Ethnocomputing: the Design and Assessment of Culture-Based Learning Software for Math and Computing Education.

By

William Edgar Babbitt

A Thesis Submitted to the Graduate

Faculty of Rensselaer Polytechnic Institute

In Partial Fulfillment of the

Requirements for the degree of

DOCTOR OF PHILOSOPHY

Major Subject: Multidisciplinary Science

Approved by the
Examining Committee:

_____

Dr. Ron Eglash, Thesis Adviser

_____

Dr. Mukkai Krishnamoorthy, Thesis Adviser

_____

Dr. Audrey Bennett, Member

_____

Dr. Bruce Piper, Member

_____

Dr. David Spooner, Member

Rensselaer Polytechnic Institute
Troy, New York
November 2014
(For Graduation December 2014)

ProQuest LLC.
789 East Eisenhower Parkway
P.O. Box 1346
Ann Arbor, MI 48106 - 1346

CONTENTS

LIST OF TABLES

LIST OF FIGURES

ACKNOWLEDGMENT

There have been so many people that have helped to further this work over the past 4 years; it is very likely that I might inadvertently leave someone out of this heartfelt thank you. Please know that if I do accidently fail to include you – I apologize in advance, and that your efforts on my behalf were greatly appreciated at the time as well as now.

First, I would like to thank my family. My wife Nancy of almost 20 years and my sons Matthew and Joshua have had me in their lives a lot less since I started this project. Once this work is complete, I will need to begin to repay them the debt of time that they are owed.

I would like to extend a heartfelt thank you to my advisers, Professors Ron Eglash, and Mukkai Krishnamoorthy. If it were not for Moorthy, I would never have applied to RPI and without Ron, there would not have been an opportunity for me to apply to. Both have been so generous with their time and patience, for which I am truly grateful.

I am also tremendously grateful to my committee, in addition to Moorthy and Ron, Dr. Audrey Bennett, Dr. Bruce Piper, and Dr. David Spooner. Their patience and understanding as this project has evolved over the past year has been very much appreciated.

I would also like to extend a thank you to Dr. Mary O'Keeffe. Mary has been a longtime friend to me and my family; she has inspired me to be the best that I can be, through her example. Mary and Moorthy are advisers to the Albany Area Math Circle, and they both have been a source of inspiration to me.

I would like to say a heartfelt thank you to Dan Lyles and Simon Ellis. Dan and I entered RPI as GK-12 fellows at the same time, and collaborated on numerous projects, conference presentations, and middle school activities. Simon has been a reliable friend, study partner, and sounding board through numerous computer science classes and all of my projects while at RPI.

I would especially like to thank all of my Ghanaian research partners. Without Mr. Gabriel Boakye, my Adinkra expert, and Mr. Richard Bonsu, my Kente Cloth expert,

this work would not have been possible. I would also like to thank Mr. Enoch Bulley, our research collaborator at the Ayeduase Junior High School in Kumasi. Without Enoch's invaluable assistance over the years, many projects would not have been possible.

I would also like to thank Linda Carey, science teacher at Hackett Middle School, in Albany, New York. Linda's willingness to stay after school and try new and interesting teaching methods with her science club students made a great deal of this research possible.

Michael Lachney has been a research partner and friend through most of this work. We traveled to Ghana together during the summer of 2014, providing support to each other's work while there, as we faced many challenges. We endured, together, through work challenges, armed robbery challenges, Ghana National Police challenges, and the resulting trauma and aftermath challenges, all with our usual good humor and camaraderie. Thank you, Michael.

On the subject of our Ghana challenges, I would like to express a heartfelt thank you to President Jackson and Provost Hajela of RPI, as well as the administration of KNUST for their expert handling of our armed robbery and its aftermath during our stay in Ghana. They, as a group, collectively 'picked us up, and dusted us off', and helped us complete as much of our planned research as possible.

I would also like to extend a thank you to the National Science Foundation's GK-12 program, Rensselaer Polytechnic Institute, and my many donors. NSF GK-12 generously funded my first two years of graduate work at RPI, along with my first two research trips to Ghana in 2011 and 2012. I received RPI TA support for the 2 ½ years following the NSF support. My third trip to Ghana in the summer of 2014 received crowd-sourced donation support through an effort conducted via 'gofundme.com'.

ABSTRACT

The United States has a serious problem in Science, Technology, Engineering, and Mathematics. The STEM disciplines are suffering from a 'Quiet Crisis'[1]. The problem is that African Americans, Latino/a, Native American, and other ethnic minority students are choosing careers in the STEM disciplines at lower percentages than their white and Asian counterparts [2]. We refer to this disparity as underrepresentation. This work focuses on the use of the Culturally Situated Design Tools (CSDTs, http://csdt.rpi.edu) as a means to counter this underrepresentation. The programmable set of CSDTs represents the development of this software from being focused on ethnomathematics, to ethnocomputing. Ethnocomputing is the reproduction of cultural artifacts in simulation. The CSDTs are part of the constructionist genre of programmable software that seeks to teach computer science concepts to students as they construct these cultural artifacts. Development work on the programmable CSDT software has provided the opportunity to examine the challenges that occur in cross cultural software development using the Agile method. This work includes ethnographic user and developer stories that have informed the development of the pCSDT software. Among the challenges of developing this software has been negotiating the controversy of what we have come to term "The Content Agnostic Position". This is the notion that all software with objects in simulation, regardless of what those objects are, possesses equal value in teaching students. This position, however, often results in the commercial or violent colonization of user spaces by corporate produced media objects and simulated gun violence. It is our view that this colonization can be diminished through the use of cultural objects in simulation, such as those found in the CSDTs. Our work is intended to create a decolonized space for mathematics and computing education. This work also reports the findings of a quasi-experiment conducted with junior high school students in Kumasi, Ghana, West Africa during the summer of 2014.

1. INTRODUCTION

This dissertation examines the educational potential of 'ethnocomputing'—a body of research that examines the math and computing ideas and practices embedded in cultural activities. By developing simulations of these cultural practices in the form of pedagogical design tools, we can create new opportunities to better engage underrepresented students. In the context of the United States, 'underrepresented' refers to African American, Latino, and Indigenous (Native American, Native Alaskan, and Pacific Islander) communities. These groups have relatively poor quality of life indicators in almost every metric: lower income, lower life expectancy, and higher rates of teen pregnancy, incarceration, and so on. Their lower rates of participation in Science, Technology, Engineering, and Mathematics (STEM) careers are both a symptom and contributing factor of this disparity. In addition to humanitarian concerns, there are negative consequences to the US economy; so much so that the problem has been referred to as the 'Quiet Crisis' [1]. Research presented in this dissertation indicates that an ethnocomputing approach to education may not only offer the possibility of improvement in underrepresented student STEM performance and interest, but also offer advantages for all students—internationally as well as in the US—in what we have termed the 'content aware' approach to constructionist pedagogy.

Regarding the specific case of U.S. underrepresented students, it should be noted that some of the causes of this underrepresentation are beyond our control as designers of pedagogical tools. Lack of resources for inner city schools, poor choices in diet and nutrition, and urban gang violence are well beyond our control. However, the ethnocomputing approach can directly address other factors. The role of the myths of biological determinism and cultural determinism—as I will detail in chapters 1 and 2—fall well within our reach. Similarly, poor pedagogical techniques resulting from 'instructionist' or 'drill and kill' approaches can be replaced by ethnocomputing's constructionist alternatives, supporting inquiry or discovery learning, as described in chapter 3. It is this combination of a culturally situated understanding and a conceptually open media that makes ethnocomputing uniquely suited to address these social issues. As

we will see, prior work described in the 'culturally relevant' teaching literature tends to use instructionist approaches. When these are replaced with constructionist methods, culture tends to drop out, and this 'content agnostic' approach often allows for a 'colonization' of learning spaces with corporate commodities—hence the ubiquity of projects in Scratch and similar platforms based on Barbie, Halo 3, McDonalds, and other commercial enterprises, as we will see in chapter 4.

Chapters 5 and 6 describe our efforts to develop and evaluate a 'content aware' alternative to 'content agnostic': one that combines the 'openness' of general programming platforms with the cultural specificity informed by research in ethnomathematics and ethnocomputing. In the academic world, math and computing are neatly separated into disciplines, courses and journals, but of course, the real world makes no such distinction. Modeling the formal systems underlying cultural designs—textiles, artistic practices, adornment, and so on—requires both forms of analysis. Understanding the interior cognitive world of these artisans, translating that via HCI principles into an interface that satisfies both children's learning needs and teacher's curricular demands, and evaluating the results with both quantitative and qualitative assessments, requires additional synthesis from disciplines such as psychology, anthropology, and science and technology studies, as well. Despite these complexities, I believe the data shows that the form of ethnocomputing pedagogy that we have introduced through our Culturally Situated Design Tools (CSDTs) brings culturally specific advantages for U.S. underrepresented groups, they address the problem of the 'commodification' of learning for U.S. majority groups, and they even show promise internationally as evidenced by our field work in West Africa.

1.1 STEM and Underrepresentation

As noted above, the lower rates of STEM participation by African American, Latino/a and other groups in the US has both humanitarian and utilitarian dimensions. In terms of its humanitarian outlook, the situation has parallels with the civil rights era of the 1960s, where activists protested the lack of voting rights for legal citizenship. In our era, the problem is a lack of 'technological citizenship'. Just as one cannot exercise their full rights as citizens without the vote, one cannot participate fully in a highly technologized world without sufficient STEM education, regardless of profession.

Avoidance of STEM fields of study can contribute to the lower rates of income in these communities. One of the most important decisions a young person can make is their choice of career. Adult lifetime earnings potential, and quality of life as determined by socioeconomic status will have largely been determined, once a student settles on a career choice, presuming the student stays within that field of choice.

In terms of the utilitarian employment outlook, underrepresentation of African Americans, Latinos, and others—along with women of all ethnicities—has a profound effect on the U.S. labor pool, by reducing the available candidates for STEM employment. In addition, scholars such as Joseph Graves [3] have pointed to connections between poor STEM education and higher teen pregnancy rates, higher rates of HIV transmission, and other health problems in the African American community. Thus, when we consider the effects of portions of the US population choosing to avoid STEM careers, it becomes clear that the cost of doing nothing in motivating and training students to be successful in STEM careers is just too high. As Jackson observes, to address this shortfall, we must be able to recruit STEM trained employees from the entire population [1]. It is vital to the economic interests of the United States that all U.S. students view the STEM fields as desirable and attainable career options.

When asked about career plans, underrepresented students often state their ambition in sports or entertainment. While it is true there are a few highly paid sports and entertainment professionals, the sad fact is that students aspiring to these careers are far more likely to fail in their chosen profession than succeed, as very few positions are available as a star athlete or entertainer. Moreover, these ambitions are used to justify neglecting one's academic success.

On the other hand, for a student that sees a STEM field of study as a viable career choice, the impact on poverty can be profound. Choosing STEM greatly increases the lifetime earnings potential of the student, which in turn effects family members and significant others that are connected to that student. A STEM choice would have a positive impact in alleviating a cycle of poverty that so many underrepresented students experience.

Succeeding at countering STEM avoidance in career preparation and aspiration would yield benefits that would extend beyond the individual STEM professional.

Students choosing STEM receive training in the Scientific Method, which would have a positive effect on those around them. These professionals would form the basis for a community that is both STEM informed and STEM confident.

Trained professionals, as STEM role models, can transform the wider community. This modeling of the benefits of a STEM education would encourage others to reconsider their own STEM options. As the community grows in its STEM inclusiveness, benefits would extend beyond the effects of educational role models. This expansion could have positive impacts on reducing the rates of teen pregnancy, which frequently robs young women of their ability to complete an education by adding overwhelming family responsibilities to their already difficult lives [3].

A STEM inclusive community may also have lower rates of HIV infection, through its knowledge of the causes of that disease, and the risks of unprotected sexual activity. Knowing the causes and risk factors of HIV would encourage young people to stand up against peer pressure to engage in those risky behaviors. STEM awareness would also assist health care professionals in guiding patients to make healthier lifestyle choices. In addition, STEM awareness would result in higher vaccination rates, which would increase positive health outcomes in all age groups.

Choosing to work in a STEM related field could also reverse internalized pathologies resulting in a healthy self-identity. This positive self-identity would be derived from the esteem of pursuing a successful career in a high value, high demand field, being a positive role model in the community, and experiencing the financial reward of high paying employment. A STEM career would, therefore, result in a producing lifestyle instead of a consuming lifestyle.

It is clear that our society needs to be able to recruit STEM professionals from all ethnic groups, not just white and Asian students. It is also clear that those students who choose STEM careers have a positive effect on their communities in addition to the personal benefit that they receive from the STEM choice. The causes of underrepresentation can be complicated and are different for each individual, with many causes being out of our reach. There are some causes of underrepresentation, however, that are uniformly pernicious and within our ability to address, such as the myths of genetic and cultural determinism.

1.2 Myths of Cultural and Genetic Determinism

The myth of genetic determinism is the roundly debunked notion that one's racial genetics—that is the genetic component specific to a geographic ancestral group such as African or Native American—predetermines a person's ability to succeed. Race has little genetic basis: humanity arose relatively recently as a single species in Africa and its genetically geographic adaptations such as skin color are even more recent. One recent study [4] shows that white skin did not arise in Europeans until about 7,000 years ago. Thus, what we think of as deep genetic differences are, in fact, a trivial component of our otherwise nearly identical genome [5]. Unfortunately, the concept continues as a way to classify and sort people: who are the most intelligent, most spiritual, and most human, and which race is best, and which is not. The ideas and definitions related to the terms of race and racism have evolved and changed over time. Far from a science that predicts human intelligence based on DNA, the racial myths of genetic determinism are a means to exert economic and political power of one group over another [6].

The myth of cultural determinism is the idea that 'authentic' participation in a culture prohibits certain activities. For underrepresented youth, this can have a negative impact on educational motivations: if there is peer pressure to see academic success as being a 'sell out' then academic failure is 'keepin' it real'. With African American students, this often plays out as an accusation of 'acting white' [7]. Fordham [8] and Ogbu [7] document the ways some African American students experience a perceived expectation among their peers to choose between their black identity and high academic achievement. Fryer and Torelli [9] found statistical evidence supporting the contention that high-achieving African American students can be, and often are accused of "acting white" by their peer group. When young students internalize cultural determinism, it acts in much the same way as this negative peer pressure, diminishing their educational potential.

Just like cultural determinism, the results of genetic determinism are at their worst when young students internalize these ideas. This internalization results in a 'self-fulfilling prophecy' that serves to limit achievement, placing the study of certain academic subjects out of the students reach [10]. Expressions such as 'I'm not good at math because I don't have the math gene' or 'I'm not good at science because my parents

aren't good at science', demonstrate how insidious these fallacies can be. All of these attitudes play directly into the social construct of race that serves only to control and limit opportunity for those in oppressed ethnic groups.

The control structure that is race leads to the disparate treatment of people of different races, which we refer to as racism. Not surprisingly, different groups define the term racism differently, determined by their experiences. Many white people understand the term racism to be referring to the prejudice of a single individual, based on ideas and feelings of dislike or hatred, while non-whites often have a better grasp of its social dimensions, as an institutional feature of society where the dominant group (whites) maintain a position of privilege and advantage over non-whites [11].

Institutionalized and systemic racism permits the maintenance of white privilege, visible to non-whites in many ways in their everyday lives, but often invisible to whites. This system of white privilege is especially evident in the values communicated to non-whites in the educational system. The experience for non-whites as they receive their education in schools indicates to them that what they perceive to be their group identity is inferior to others [12].

There is a correlation between race and IQ, and therefore, race can be a predictor of IQ scores as well as the likelihood of academic success; however, this is due to the social and economic consequences of systemic racism and its consequent historical harms, and not any genetic effects. For example, the low IQ scores of Jewish immigrants in the 1920s were due to their lack of familiarity with the U.S. language and customs [33]. IQ scores and intelligence testing, in general, are not reliable measures of student's ability, but rather they are a measure of the experiences their economic status in society has allowed them to purchase. As a consequence of the Flynn effect [13], black IQ scores have risen an average of three points per decade since 1930; they are now well above the white average from that era [14]. East Germans and West Germans are genetically indistinguishable, but their IQs were dramatically different until reunification [15].

In conclusion, while race and culture are not inherent barriers for underrepresented students, the lingering myths reinforce the barriers keep underrepresented students from higher academic achievement. Ethnocomputing offers a means to oppose these myths, transforming culture from barrier to bridge.

1.3 Research Questions

Prior work in ethnomathematics and ethnocomputing shows strong potential to counter the myths of biological and cultural determinism. Traditional practices (e.g., weaving, beading, sculpture, tattoo, drumming, and graffiti) show mathematical concepts such as Cartesian and polar coordinates and transformational geometry, and computational practices such as iteration and conditionals. Creating educational software that allows students to utilize these simulations in order to understand this traditional knowledge and then translate it into contemporary classroom skills had already achieved some initial success before I began my dissertation. However, the reality of creating this transformation from indigenous knowledge to classroom software is fraught with complexities, and these complexities are what informed my research questions:

1) Modeling the formal systems that underlie cultural designs (e.g., textiles, artistic practices, adornment, and so on) cannot be accomplished simply by analysis as an outsider. Developing methods for interviewing artisans and understanding their interior cognitive world are major areas of investigation in this work.
2) Once we have those formal models, translating that into a user-friendly interface constitutes a second area of investigation. This is not simply a matter of applying known HCI principles, since it must satisfy both children's learning needs and teachers' curricular demands, and at some level still have sufficient fidelity to the artisan's internal cognitive model. Another issue is that the software has to exist within the schools' technological ecosystem of computing machines and networks: security restrictions and obsolete equipment can often prevent otherwise optimal solutions.
3) Locking down the design tools in order to allow only a limited set of 'authentic' cultural productions is one extreme I wanted to avoid. However making it too 'content agnostic' invites the kind of commercial colonization that we show in the Scratch community in chapter 4. Thus, another area of inquiry involved finding the right balance between these two extremes. This came in the form of

'programmable' CSDTs, ("pCSDTs"), which was my major contribution as a developer on this project.

4) Formative feedback informed a gradual evolution in the interface design through an "Agile" model of iterative development.
5) Summative evaluation was used as a final check on the viability of the software in meeting our goals of improving the math and computing performance and interest of underrepresented students, as well as for students in a 'postcolonial' site in West Africa.

These five areas of research defined the scope of investigation for this dissertation.

1.4 Cultural Capital

If cultural practices are to be incorporated into educational software, we need to consider ways to investigate the changing value of those practices as they move across different media, the 'ownership' of them, their varying levels of authenticity depending on the identity of the maker, etc. An important concept for framing these questions is that of Bourdieu's idea of cultural capital. Cultural capital, as Bourdieu defined it, is the set of cultural practices that come with one's identity as a member of a particular social class [16]. These practices often confer benefits, often just a matter of knowing how to act in certain social situations. His theory was developed when he investigated why poor or working class French students failed to get upper class jobs. He found that knowing obscure bits of knowledge (e.g., the cultivation of a taste for certain kinds of food, music, art) are instilled in children, and that these behaviors act as gatekeepers throughout their lives.

Ethnocomputing attempts to stand that relationship on its head: after all, children who are part of a family or community of a lower socioeconomic status still have cultural capital in their own heritage arts and vernacular practices. Rather than discard that as irrelevant for its inability to allow them to 'pass' as upper class society members, we instead investigate its value as 'computational capital'—a well spring of algorithms, geometric forms and other formal properties. Just as financial capital can be made more liquid or fungible when converting real estate to cash, the simulations of CSDTs can

make the culture of these underrepresented students more fungible as forms of computing and mathematics.

Framing math and computer science as culturally situated with the CSDTs, returns to students what had been stripped away. This is especially rewarding for me, when we looked at this process from the point of view of a students' culture imparting to them some advantages, frequently from parents who look to improve their children's lives with the benefit of what they themselves have learned. The difficulty for underrepresented students, with only a small number of exceptions, is that their parents did not successfully navigate the public school system, and yet these parents are responsible for seeing that their children accomplish what they did not. In the privileged world, this is not the case, since these parents are generally able to provide an enormous amount of guidance to their children. Privileged parents know what to purchase for their children and they have the means to do so, yielding all of the requisite experiences they need to know and have, in particular, those that are most beneficial and helpful to academic success. The Culturally Situated Design Tools help to restore to students some of the value of their own cultural capital – value that has been there all along, but unacknowledged.

1.5 Measuring Success in Computational Thinking

The assessment of K-12 computing education can be straightforward when it comes to vocabulary or other materials that can be tested on a multiple choice exam, but the knowledge gained through programming experience is not as easily determined. As Resnick states the problem, "there is little agreement about what computational thinking encompasses and even less agreement about strategies for assessing the development of computation thinking in young people [17]". One complication is that there may be a series of incremental improvements, with a great deal learned, despite never reaching the original goal. It will be important to capture these incremental successes in the natural order in which they occur, so as to track the development of computational thinking in students as they work to problem solve their way through the development of their designs.

In their work with the Scratch programming simulation environment, Resnick and Brennan have attempted to assess student learning through three different methods: 1) project portfolio analysis, 2) artifact-based interviews, and 3) proposed design challenges [17]. Project portfolio analysis attempts to measure the number of code blocks used in the creation of a student project, trying to get at the depth of exploration of the available code block tools offered in the Scratch programming environment. Artifact-based interviews involve talking with students about their Scratch project. Design challenges involve asking the student to select from a pre-determined list of tasks that are designed to demonstrate the depth of their knowledge of programming in Scratch.

In working to assess the efficacy of teaching computing with the pCSDTs, an analysis method of student successes must be developed that accounts for the various nuances that indicate success. For that reason, we used a mixed-method approach, one that includes both quantitative data and qualitative student observations and interviews, in order to improve the assessment. Again, following Resnick, it is important to develop a "computational thinking framework" to guide these assessments [17].

1.6 Software Development

This dissertation focuses on the use of CSDTs as the embodiment of the ethnocomputing concept. The CSDTs are a set of Java and flash applets, freely downloadable from the Internet. Each of the CSDTs focuses on a different craft or cultural practice, typically one linked to the heritage culture or vernacular culture of underrepresented students. Ethnocomputing research begins with the search for math and computing concepts and practices embedded in these crafts, and develops CSDTs as a medium in which their simulation can make use of these indigenous concepts. The CSDTs are, however, sufficiently open-ended to allow flexibility in the creative agency of students as they develop these designs, which often results in hybrid forms, as students bring their own sensibilities, experiences and imaginations to bear on their creations.

This work afforded the opportunity to investigate cross-cultural software development, by means of ethnographic studies of craftspeople, student users, and developers. The CSDT software development followed the Agile method, where the developer starts with a tentative solution and then refines that solution over time, until a

satisfactory final version results. The revision process employed feedback from craftspeople, the students using the software, and teachers in their classrooms.

The CSDTs are part of the constructionist genre of programmable software that seeks to teach mathematics and computer science concepts to students. Constructionist learning theory states that student learning occurs through constructing artifacts, in this case, simulations of culturally situated artifacts within the CSDT software. During this construction, there will be a 'dance of agency' [18], [19] that occurs as the student negotiates their way through using the software interface. Occasionally, this dance will create errors or expose misconceptions, but as the student continues to work, these all tend to get resolved in the production of the final artifact.

1.7 Content Agnostic Position

Many proponents of constructionist based learning state that the simulated content is irrelevant. Whatever the topic or subject of the students efforts, *ceteris paribus*, the constructionist learning results are equally positive outcomes. For example, the Scratch programming environment takes this approach: the subject of the Scratch simulation is entirely up to the student. However, as we will see in chapter 4, these content agnostic platforms frequently result in simulations that reflect a consumption-obsessed culture, which is antithetical to a healthy cognitive development for children.

On the other hand, a medium which restricts children's creativity too much would also be counterproductive. As we will show, it is possible to have the best of both worlds: a 'content aware' medium that lets students begin in a design realm that honors authentic, grass-roots practices (which may be urban cultural practices such as graffiti, as well as heritage practices such as weaving textiles), but also one that can be adapted through the student's ingenuity and creativity and shaped to any topic or content. The preliminary data from this research shows that this approach is sufficient to satisfy both the need for healthier subject matter and the children's own desires to deploy pop culture references or other ironic or playful juxtapositions.

1.8 Conclusion

In this overview, we have examined the significance of the problem of lack of STEM participation by underrepresented students, its roots in the myths of biological and cultural determinism, and the potential for ethnocomputing to re-envision this cultural identity as a bridge rather than a barrier. We have briefly reviewed the process for ethnocomputing development, the significance of cultural capital as computational capital, and finally, the tension that must be negotiated between the over-restriction to cultural form and the under-restriction of the content agnostic position, in order to achieve the ‘content-aware’ balance sought by ethnocomputing.

In closing, I note that the potential advantages of the content-aware approach covers more than just students from underrepresented groups. Anti-racist education is good for everyone. It can help the dominant culture to understand traditionally silenced 'ways of knowing', and also the people who hold onto these unique knowledge systems [20]. It is my hope that ethnocomputing can be used as a tool to decolonize computing and mathematics pedagogy. It can also be a way to approach the healing of historical harms, by reshaping mainstream U.S. culture's tendency to think with colonial attitudes concerning marginalized others' 'primitive knowledge' or 'primitive way of life'. Additionally, ethnocomputing can be a way to bridge the thought processes between groups of people who hold very different world views.

2. WHY ETHNOCOMPUTING?

2.1 Introduction

In this chapter, we look at the development of ethnocomputing from ethnomathematics. Ethnocomputing inherits a great deal from the ethnomathematics research programs that preceded it, and so it is right to begin this chapter with a review of the role that ethnomathematics plays in both mathematics education and what it passes on to ethnocomputing. We next turn to ethnocomputing and the Culturally Situated Design Tools. The CSDTs are part of the constructionist genre of programmable software that seeks to teach computer science and mathematics concepts to students as they construct cultural artifacts. The CSDTs started out as a set of Adobe Flash programs with a primary focus of teaching the mathematics concepts embedded in cultural practices. The work on the CSDTs evolved into work on the pCSDTs or programmable Culturally Situated Design Tools, a set of Java applets deployed on the Internet. From here, we look at the domains of interaction that our software faces in the classroom. These interactions represent three challenges to the software developer. 1) To create pedagogical software that is suitable for the classroom. 2) This software needs to be suitably designed to "translate" the particular indigenous or vernacular knowledge under investigation into the analogous knowledge forms contextualized for student learning. 3) Successfully negotiate between the “fidelity” of the simulation as an exact replica of the indigenous concept, and the utility of the simulation as a fit to the classroom curriculum. Finally, we look at the design process through an ethnocomputing lens for the West African crafts of Kente cloth weaving and Adinkra stamping.

Portions of this chapter previously appeared as: B. Babbitt, D. Lyles, and R. Eglash, "From ethnomathematics to ethnocomputing," in *Alternative Forms of Knowing (in) Mathematics*, S. Mukhopadhyay and W.-M. Roth, Eds., ed Rotterdam, The Netherlands: Sense Publishers, 2012, pp. 205-219.
Portions of this chapter have been submitted to: W. Babbitt, M. Lachney, E. Bulley, and R. Eglash, "Adinkra mathematics: a randomized, controlled study of ethnocomputing in Ghana," *For the Learning of Math.,* submitted for publication.

2.2 Ethnomathematics

The ethnomathematics literature has no lack of visionary statements on what its advantages might be. In some cases, the motivation comes from the concept of "cultural relevance" to a specific population. Jama [23] for example draws out normative connections between indigenous mathematics and science of the Somali culture in the Horn of Africa region and local school curriculum. He suggests that ethnomathematics can be used as a "special language" to help students see themselves as historical and political actors through deep engagement in their own cultures' mathematical heritage. In other cases, ethnomathematics is framed more broadly as a way to challenge curricular Eurocentrism [24]-[26]. The latter stresses the use of ethnomathematics not in terms of specialized fit to a particular population, but rather as a way to enable student's understanding of math as an empowering tool in the repertoire of humanitarian practices. As a research program present within and outside school walls, ethnomathematics challenges classic notions of math education while also revealing power dynamics about who is represented and hidden within curricula.

Zaslavsky [27] describes her early ethnomathematics research as motivated by the fact that African mathematics did not appear in US library catalogs, nor did she find any information on the topic when she contacted the Secretariat in Ghana. This is not a casual happenstance. Western "exceptionalism" has a pervasive hold on its math and science as the only accurate way to explain reality [28], [29]. This has profound influence on non-Western education. Indigenous math and science continues to be marginalized in Ghanaian and other African curricula [30], despite persuasive arguments that its inclusion may help with problems of enrollment, engagement, and performance [31].

In the US context, these arguments have found empirical support in the work of the Alaska Native Knowledge Network. Lipka et al. [32] for example developed a set of culture-based lessons for native Alaskan students, which combine discovery or inquiry learning pedagogy with contexts that emphasize native Alaskan traditional knowledge. Their work shows statistically significant improvement in pre/post test scores for the experimental group in comparison to their control group. Our own team has conducted research that provides similar quantitative evidence for the efficacy of this approach among students of many racial backgrounds--not only Native American but

African American and Latino as well. In this study, African fractals were introduced in an ethnically diverse high school computing class in New York City: this experimental group showed statistically significant improvement on pre/post comparisons relative to a control group which received similar instruction without any cultural connections [33].

Despite this evidence for efficacy and liberatory potentials, there has been little serious adoption in most curricula [34]; even when cultural connections are introduced, the overwhelming tendency is to only superficially represent indigenous knowledge [35a]. While the "culture" side of ethnomathematics can vary widely--including topics from vernacular culture such as graffiti, working class skills such as carpet laying, and even investigations of cultural influences in professional mathematics--indigenous math plays a special role: as noted in chapter 1, it directly contradicts the pernicious myths of genetic determinism.

Finally, there is a mimetic resonance between these historical modes of epistemological domination and pedagogical styles of authoritarian learning. Just as rote memorization is often justified to satisfy the ends of testing at the expense of learning that students find meaningful, lasting colonial legacies form a "neocolonial" context that justifies a putative universal form of knowledge at the expense of the flourishing of cultural traditions of living. Thus, the potential for indigenous knowledge to have meaningful influence on student performance is not merely a matter of test scores, since institutional bodies that aim to meet the demands of a workforce employed by global financial forces, environmentally destructive industries, and increasingly deadly militaries create those tests. It is no surprise that indigenous knowledge systems appear to be ill suited for the mathematical and computational knowledge base of these enterprises. Nonetheless, it is our hypothesis--supported by statistically significant empirical studies--that incorporating indigenous knowledge systems into a math and computing curriculum can both raise student scores on tests that are influenced by these institutions, and simultaneously help to impart the cultural, ecological and ethical knowledge [36] that will offer solutions to these harmful global forces.

A growing body of research suggests that education can improve its contributions to students' knowledge and understanding when it offers them the opportunity to interact emotionally and critically with content that is culturally relevant to their communities and

their identities [37], [38]. As we will see in chapter 6, our Ghanaian case study supports that position, using a controlled, randomized study to show curricula based on research in ethnomathematics and ethnocomputing can result in better student motivation and academic engagement. Simultaneously, we suggest that including indigenous knowledge can offer deeper connections with sustainable relations between the natural and social worlds. We see this study as part of the larger "culturally relevant pedagogy" framework that fights against political and epistemological inequalities through "restoring cultural dignity and offer[ing] the intellectual tools for the exercise of citizenship [39]".

As noted by Rosa and Orey [40], modeling is an essential tool for ethnomathematics. But when we create a model for a cultural artifact or practice, it is hard to know if we are capturing the right aspects; whether the model is accurately reflecting the mathematical ideas or practices of the artisan who made it, or imposing mathematical content external to the indigenous cognitive repertoire. If I find a village in which there is a chain hanging between two posts, I can model that chain as a catenary curve. But we cannot attribute the knowledge of the catenary equation to the people who live in the village, just on the basis of that chain. Computational models are useful not only because they can simulate patterns from the outside, but also because we can compare their underlying algorithm to the ethnographic data we gather how artisans create artifacts, describe their ideas and connect material designs with cultural concepts, thus providing insight into this crucial question of epistemological status.

2.3 Ethnomathematics to Ethnocomputing

Ethnomathematics faces two challenges: first, it must investigate the mathematical ideas in cultural practices that are often assumed to be unrelated to math. Second, even if we are successful in finding this previously unrecognized mathematics, applying this to children's education may be difficult. These difficulties give rise to the approach we refer to as "ethnocomputing."

Ethnocomputing offers two advantages over ethnomathematics. First, although we like to think of mathematics as being comprehensive in its ability to model patterns, some pattern generation systems are better conceptualized through the disciplinary idioms of computer science. Second, the conceptual framework of computing—the idea of

information processing, algorithms, graphical user interface, etc.—allows new insight into the artisans' own perspective in cases in which there is an analogous process.

As noted above, our work with CSDTs made it clear that there is a component of ethnomathematics that has received little attention, because such "computational thinking" [41] is outside the purview of the standard math curriculum. As Margolis [42] notes, computing education is a key to the high-status skills and knowledge that allows a student to tap into the grid of twenty-first-century opportunities; one where under-represented students are often left out.

2.4 From CSDTs to pCSDTs

The Adobe Flash based CSDTs were designed to teach the mathematical concepts embedded in indigenous practices and artifacts. However, this approach to math education seemed to leave out the skill sets that Margolis [42] talks about as being important to possess in academic and career success. The idea of converting the CSDTs into programmable interfaces that would teach these skill sets to underrepresented students was indeed compelling.

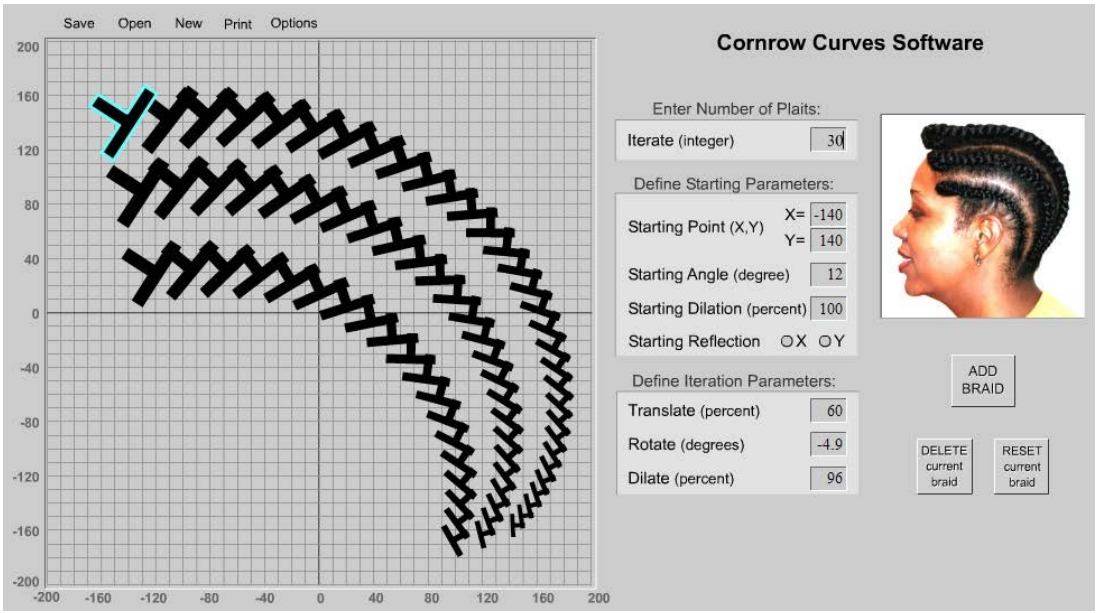


Figure 2.1: Adobe Flash version of the CSDT for cornrow hairstyles

It was unclear, however, whether this effort to teach both math and computing in the same interface could be applied to all CSDTs. Fractal geometry is a special case in that it is inherently mixing computing and mathematics. What about teaching conditionals, data structures, and algorithms? Such concepts were present in the CSDTs, but too deeply embedded in the tools. Take, for example, the "Cornrow Curves" simulation. Figure 2.1 shows the CSDT control panel and resulting simulation for three braids. The photo at right is one of many "goal images" that students can attempt to simulate. At left is the simulation. The left-most plait of the top braid is high-lighted to indicate that the numbers in the control panel refer to that braid. The simulation uses a recursive loop in which the original plait image is duplicated, and then geometric transformations are applied to the duplicated plait. This cycle is repeated, duplicating the previous duplication, until the desired number of plaits have been generated. However this algorithm remains invisible to the students; they only see input boxes for the parameters.

In order to make that algorithm visible, we would have to create a "programmable" Culturally Situated Design Tool, or pCSDT. Projects at CMU ("Alice") and MIT ('scratch") have developed programming interfaces that allow students to generate algorithms by dragging and dropping snippets of programming language ("codelets") into a 'script"—thus eliminating the frustrating experience of having a program fail because of an obscure syntax error in the code. But would students who had experienced the ease of the older CSDTs, with a purely parametric interface, be willing to create these scripts? Would we have to hide the older versions from them? Finally, we also needed to create an interface that would be easily extensible for the creation of additional pCSDT applications—we did not want to build a unique interface for each tool. And of course all this needed to happen while keeping true to the cultural connections that motivated the project in the first place.

Our design efforts crystallized around a Java applet that could be easily deployed on the web, but also brought in on physical media (CD or flash drive) in case we were in a situation with low bandwidth (or no bandwidth) internet access. To meet the requirement of being easily extensible, the program is constructed in layers. The Core layer contains the interface that is used for every programmable CSDT. The application

layer contains all the code relevant to each specific tool. For example, in the case of the Cornrow Curves applet, the application class says that we want codelets for transformational geometry such as "Rotate," a Cartesian grid for the background, a plait image for the default object, etc. Some codelets such as "Repeat While" loops are common to all pCSDTs, so they lie in the in the core class. Figure 2.2 shows the resulting pCSDT for cornrows.

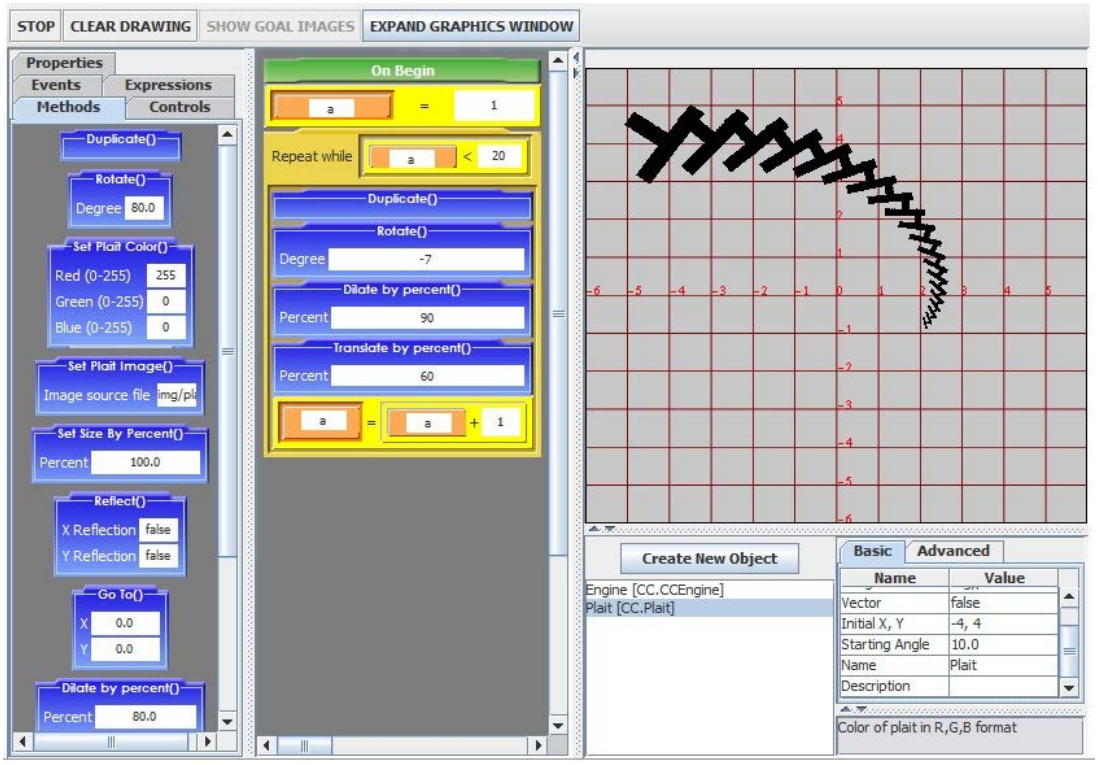


Figure 2.2: Programmable Culturally Situated Design Tool for cornrows

The panel at the left contains the list of codelets, the center panel is the script created by dragging and dropping codelets, and the right-most panel is the simulation window. When users are finished with a script they can expand the simulation window to full screen before activating the script. At the top of the script, the user has declared a counting variable (called "a") and initialized it with a value of 1. The next codelet is a control loop, to the effect of: "While a < 20, do the following." Inside the loop are codelets for duplicating the plait image, and applying geometric transformations (rotation, scaling, and translation). At the end of the loop, the variable "a" is incremented by 1. Thus the script makes visible the algorithm that was invisible to users of the

original CSDT. We hypothesize that a non-numeric version of something like this algorithm is also cognitively available to the stylists who create these braids.

One of the most interesting aspects of ethnomathematics simulations is that the results that they produce can surprise the software developers who create them; that is, we were not completely certain what visual patterns we would be able to produce until we actually created the simulation and began to experiment with it. We found that the new pCSDT allows many patterns that were very difficult to make with the old version. For example, in Figure 2.1 you can see 3 braids created on the old version: each of those braids required a separate series of trial and error experiments. In the new pCSDT, nesting one control loop inside another allows the user to automate the process of generating a series of braids. Another problem is that real cornrow braids sometimes have rotation values that switch back and forth, like a sinusoidal waveform. On the old version the user would have to create that effect by piecing together separate braids. The new pCSDT version allows users to introduce conditional codelets ("if-then" or "if-then-else") so that values such as rotation can be altered at any point in the braid.

The pCSDT version also allows some patterns that are impossible to make with the older version. For example, it is simple to introduce color by placing the color codelet into the script and entering R-G-B values (0-255). By inserting the counting variable ("a") rather than a value (and this must be the value "a" multiplied by a constant, for which there are codelets), the color can increment with each plait, such that a braid can begin with blue and end with red, with corresponding gradients of purple in-between. By introducing a second variable ("b") we can keep track of odd or even duplications, and thus alternate colors in the simulated braid (Figure 2.3). Interestingly, we later realized that alternating colors are often used in physical braids: the ethnocomputing approach helps to alert us to formal aspects that were previously overlooked, and create a model for their simulation.

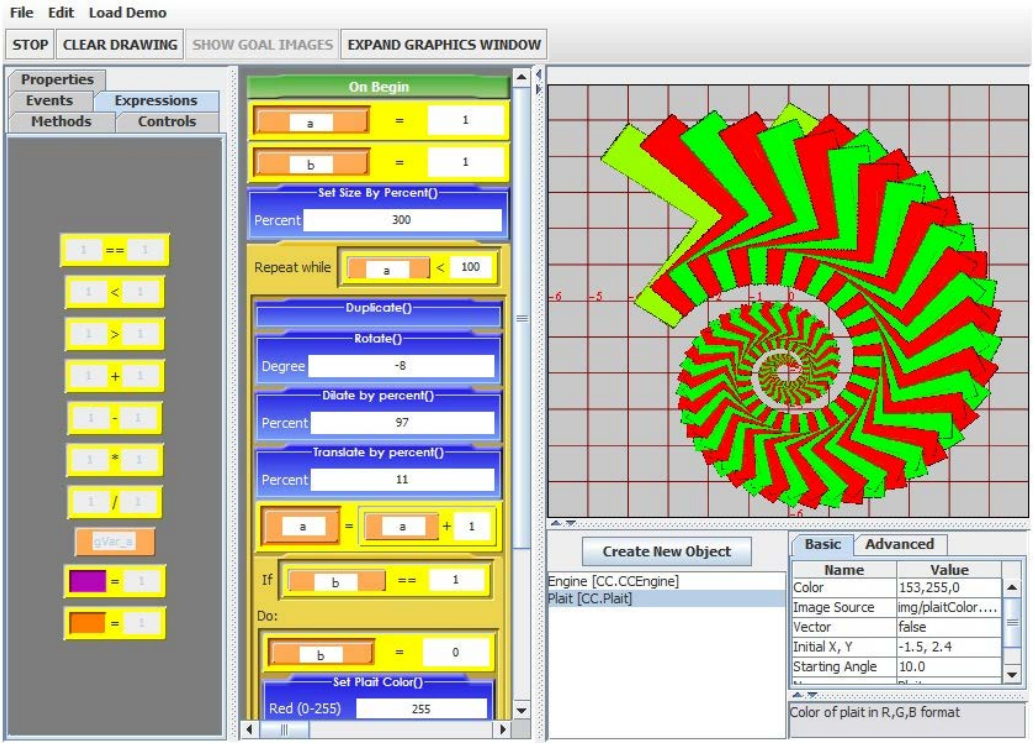


Figure 2.3: Simulated braid with alternating colors

2.5 Domains of Interaction in the Classroom

As noted above, there is a second challenge to ethnomathematics, and that is the challenge of converting these results into a pedagogical form suitable for pre-college classrooms; in particular for under-represented ethnic groups. Of course there are other applications for ethnomath, but education has been the most important. That is because of the possibility that cultural connections to math may improve the low performance and interest in mathematics that we tend to see in African American, Native American, Latino and Pacific Islander children in the US. As noted in chapter 1, ethnomathematics could potentially offer a powerful counter to both the "acting white" myth and the genetic determinism myth. But merely pointing to a photograph of some intricate basket or monumental pyramid is not sufficient for engaging children or developing their mathematics skills. Here too computing can contribute.

In summary, we find the following three domains of interaction for ethnocomputing in the classroom:

a. Simulations must "translate" from the particular indigenous or vernacular knowledge under investigation into the analogous knowledge forms contextualized for students in classrooms. For example, Native American traditions use the "four winds" or "four directions" as an organizing principle across many different knowledge systems: cosmology, religion, health, architecture, weaving, etc. [43]. This makes it an excellent candidate for teaching the Cartesian coordinate system using simulations of these artifacts such as bead work; we can clearly justify the four-quadrant coordinate system as an indigenous invention, and not merely a western idea that is imposed on these artifacts [44]. At the same time, a pedagogy that introduces these artifacts should do so with the social context in which they arise. The cultural background pages for the "Virtual Beadloom" CSDT, for example, includes the use of Iroquois bead work in their US treaties and the influence of the Iroquois confederacy on the creation of the US constitution. In such instances we have attempted to steer a path between the Scylla of "white-washing" history (such that the horrors of exploitation and oppression are completely erased), and the Charybdis of a story of "victimhood" (which could demoralize students).

b. These math and computing analogies are rarely exact; there is typically some negotiation between the "fidelity" of the simulation as an exact replica of the indigenous concept, and the utility of the simulation as a fit to the classroom curriculum. For example, weavers have to worry about fitting horizontal weft threads into the vertical warp without creating slack, hiding the ends when a new color is started, etc. All of these activities might be modeled: for example you could model the relations between two adjacent horizontal weft strands as 180 degrees out of phase. But that would complicate its use at lower grade levels where the concept of "phase" is not taught, and even at upper levels, being forced to think about phase while simultaneously using an iterative algorithm would make use potentially frustrating. In contrast to critics who complain that ethnomath adds too much external math to artifacts, the challenge in developing these simulations is to leave out much of the "high fidelity" modeling that would potentially be possible, in order to create a lower fidelity model that is both optimal for use and offers a clear translation of indigenous knowledge.

c. In addition to attempting to negotiate the tension between fidelity to the indigenous conception and utility to the curriculum, a third tension exists when trying to satisfy student needs for relevance and creative initiative. For example, in our initial attempts to use fractal models of African artifacts [45], we found that African American students occasionally expressed some hesitation over what was, for them, dusty museum objects. For this reason, our first simulation focused on cornrow hairstyles, which offered a compromise between African heritage and objects and practices familiar to them as part of contemporary African American culture. However as our websites have developed we have found that even for familiar practices (bead work in Native American communities, graffiti among the urban "underclass," etc.) there is a need to teach these histories (where else are they going to learn about the history of graffiti?). Similarly, the ability to make creative use of these tools, and generate their own designs (some of which bear no resemblance to traditional examples) is critical for engaging these students, and encourage a sense of ownership over the mathematics. Moving from consumption to production, taking pride in self-efficacy and designs, learning to use math and computing as a means of self-expression rather than the disciplinary regime of "you got the wrong answer"—these are all critical components of ethnocomputing pedagogy.

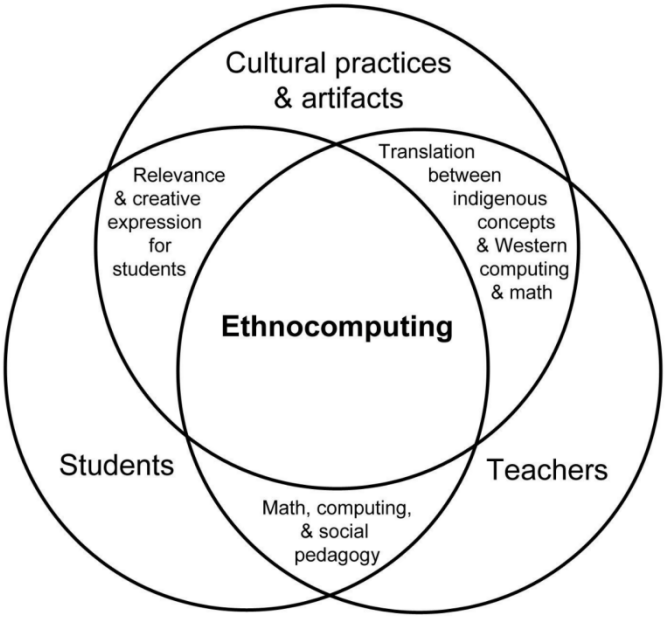


Figure 2.4: Three domains in ethnocomputing

2.6 Inquiry Learning

Inquiry-based learning, in which students either invent a question themselves, or have their inquiry assigned to them, is increasingly supported by innovations in pedagogy (cf. [46]). A crucial component of the education theory supporting inquiry learning is that of scaffolding: here some temporary conceptual aid allows a student to advance their understanding, such that with a firmer grasp on new concepts, they can then climb to higher levels. Brush and Saye [47] introduce the terminology of "hard" and 'soft" scaffolding. They refer to teachers as providing 'soft" scaffolding, by which they mean it is contingent and adapted to circumstances. In contrast, they suggest that multimedia systems, of the type they introduce (which consists primarily of hyperlinked media to support high school social studies inquiry), can be labeled "hard scaffolding" because the designer must pre-plan whatever learning aids will be available.

In our case, neither category fits well: the scaffolding is contingent, not pre-planned: we never anticipated that a student would generate a braid simply by creating each plait as a separate object, as we describe below. But it is also not a contingency generated by a teacher; rather it is a contingency generated by the interaction between a student and a digital medium that is sufficiently flexible and powerful enough, to allow creative explorations.

Rather than call this hard or soft inquiry, a better category might be "mangled inquiry." Student's struggling with writing a script can be described using Pickering's [18] model of "The Mangle," in which he detailed how scientific discovery occurs as a "dance of agency." Pickering describes the failure to accomplish a particular goal as "resistance" (in the language of Pickering, nature resisting a "capture" of its agency by some model or machine). The scientist then responds by seeking a new strategy to overcome that failure—changing models or machines or procedures until s/he finds one that works ("accommodation"). In participant observations, we have seen this play out in placement of the codelets. For one student, the resistance came in the form of a logic error of the student placing a "Duplicate" codelet outside the control loop in their script. The resulting struggle to find a solution eventually led to accommodation when they moved the codelet inside the loop. However, it is critical to understand that in Pickering's view, there is not simply one "correct" model or machine. Multiple different

accommodations are possible, including a change of goal. And in fact, there are multiple locations within the script that would have allowed this student to successfully generate a braid, although the behavior might have varied slightly (e.g. there would be one less plait if you duplicated after the variable is incremented). In a different student observation, it was clear that this mangled inquiry occurred in two stages: first when the student attempted an initial strategy that was temporarily successful, creating the braid with individual objects, and second, this allowed the student to proceed to the point where they were able to set a higher goal for themselves (from the goal of merely making a braid by any means necessary, to the goal of having the script automatically generate the entire braid sequence).

Indeed we can view our own attempts through this same lens of "mangled inquiry". Our planned evaluation system at a local school met with initial resistance; there was no way to use pre/post evaluations given the enormous variation in student attendance. But we accommodated that resistance by focusing on the discussions that followed the software experience, and thus gained some insights into the elements that increased students' engagement in math and computing conversations.

Inquiry learning works best when it is open-ended. Students need to be able to ask questions, pose answers, and explore the implications of those answers—not necessarily "the one right answer" but rather discovering what new patterns emerge when those answers are used. In that exploration, new questions can then be developed for further consideration. The new pCSDTs offer exactly that scenario. As drawings are created they can be changed by adding additional coding elements. This added complexity will result in scripts that could benefit from rewriting, and the results offer new horizons for further exploration. As Resnick et al [48] note about MIT's Scratch, it is critical to offer a "low floor" (easy to get started) and "high ceiling" (enormous room for expansion).

2.7 Ethnocomputing: Kente cloth

2.7.1 The History of Kente Cloth

The history of Kente cloth weaving is not completely certain, however, according to Rattray weaving Kente was probably introduced to the Akan people of Ghana around the 1600's. Tradition has it that Ota Kraban returned from the Ivory Coast with a loom

and set it up at Bonwire, Ghana [49]. Early Kente cloth was woven from silk thread, however, today the cloth can be produced by any number of different materials which is often determined by its commodity price, as well as the desired look of the finished product.

Kente Cloth patterns can be observed in the making in many places in Ghana, particularly in the village of Bonwire, and is offered for sale through the major markets of most Ghanaian cities, such as those of Kumasi and Accra. The distinctive colors and patterns are found in everything from sandals, hand bags, shirts, and of course cloth. In the United States, Kente Cloth makes an appearance in many different ways, but usually as a means of conveying a symbolic message to African Americans of the African diaspora.

Figure 2.5: Ohene Anewa [50]

Figure 2.6: Nkyimkyim [50]

Figure 2.7: Afa [50]

Examples of Kente cloth fabric can be seen in Figures 2.5 – 2.7. Each of these designs convey a particular meaning through the design of the geometric figures. Ohene Anewa (Figure 2.5) represents the Kings 'All seeing eye', as in the King knows all. Nkyimkyim (Figure 2.6) is a zigzag pattern which represents the notion that life is not a straight path. Afa (Figure 2.7) means 'I have taken it' [50].

2.7.2 Kente Cloth pCSDT

The idea of creating a Kente cloth simulation came about while conducting research on Kente in Ghana. Kente cloth is often mentioned in passing in the ethnomathematics literature, but there is little sustained discussion. When we think about the algorithms involved in the creation of Kente, it becomes easier to see that computing

offers a rich array of possibilities. Software design requirements would need to illuminate just how the weaving process should be represented to the user. Once this representation is determined, it should then be straight forward to create the weaving simulation. The only remaining decisions would involve determining the level of complexity to hide or show to the user in the codelets for the simulation.

The geometric patterns of Kente, along with the vibrant use of color in the designs, make it especially suitable in an ethnocomputing context. However, simulating Kente Cloth would not be without lengthy design negotiations in order to maintain fidelity to the practiced craft, and suitable design for use with students. In chapter three we will look at the process of software development related to our Kente cloth simulation efforts, and in chapter four some observations of children using and learning with the Kente cloth simulation tool.

2.8 Ethnocomputing: Adinkra Stamping

2.8.1 The History of Adinkra Stamping

Adinkra symbols can be primarily observed today in Ghanaian textiles. The Akan peoples of Ghana adopted Adinkra textiles around the year 1800, yet the origins of the craft remain uncertain [51]. Many of their geometric forms exist in older archaeological artifacts, across a wider geographic range. In the case of the textiles, these were originally used in the funerary arts with each symbol communicating a particular idea to the departed loved one. Contemporary uses of Adinkra symbols have expanded well beyond the funerary arts. Traditional Adinkra artisans in Ntonso, Ghana still carve symbols from the calabash gourd, make their own ink, and stamp various types and styles of cloth primarily for tourists that visit their shops. A drive through nearby Kumasi reveals Adinkra symbols painted on garden walls, the columns of Internet cafes, and molded into the backs of plastic chairs. In the 21st century, Adinkra has become a global phenomenon. In the United States, Adinkra symbols adorn everything from t-shirts and jewelry to braiding salons, and their forms and names are used in lively and creative ways by African American community organizations and hip-hop artists.

2.8.2 Adinkra Stamping pCSDT

Figure 2.8: Dwennimen

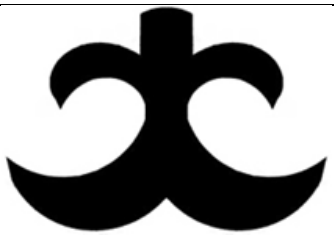


Figure 2.9: Akoko Nan

Figure 2.10: Sankofa

Willis [37] offers a comprehensive summary of the meaning assigned to each Adinkra symbol. Dwennimen (Figure 2.8) is the ram's horns and means strength in mind and body, but also means humility. Akoko Nan, the hens foot (Figure 2.9) is the symbol of protectiveness, parental discipline, but also patience and mercy. The Sankofa (Figure 2.10) means 'go back and get it', also 'go back to your roots', generally: learn from the past in building the future [51].

Our efforts in constructing an Adinkra pCSDT have centered on the geometric shape of the symbols themselves. As is apparent in Figures 2.8 through 2.10, many Adinkra symbols incorporate logarithmic curves in their design. This is perhaps not surprising, as the logarithmic curve has been described as 'the curves of life' [52], and many of the Adinkra symbols are rooted in physical, sometimes living objects such as the ram's horns and the hens feet above. Clearly any attempt to model Adinkra in a pCSDT would involve simulating logarithmic curves, the question becomes how and to what extent the complexity of the mathematics is revealed to the student using the software. Please see chapter three of this work for development stories concerning our Adinkra stamping simulation development. Please see chapter four of this work for participant observations of the Adinkra stamping simulation.

2.9 Conclusion

In conclusion: the answer to 'Why Ethnocomputing?' that this chapter has presented is three fold.

First, ethnocomputing extends ethnomathematics in important ways. Ethnocomputing seeks to include constructionist ways of learning that encourage 'computational thinking'. The important ability to observe a process, determine how and if the process is working as expected, and make changes to remove imperfections [41]. The skill of computational thinking is thus more important for students to develop than simply competing in the job market, although that too is critical.

Second, ethnocomputing seeks to deliver these lessons using 'inquiry learning' methods. Inquiry learning, with all of its varying degrees of exposed scaffolding and complexity, soft inquiry, hard inquiry and the like, lets the student explore for themselves at their own level of expertise. This exploration allows students to build their own knowledge, and most likely, because of the deep interaction with the software during this construction, they will retain the knowledge that they build.

Finally, and in my opinion, most importantly, ethnocomputing – through its choice of cultural practices and artifacts in simulation – exposes students to the cultural capital that they have always had in some sense, and yet it is unlikely that they ever tapped into the full extent of its power. This contextualization of learning in cultural objects helps students, especially underrepresented students, own their learning. This has been the most rewarding part of this research for me, the experience of having a student 'light up' at the notion that someone took the time to design and build a simulation of her hair style.

3. SOFTWARE DEVELOPMENT AND HCI

3.1 Introduction

Ethnocomputing requires that we focus in on the various aspects of 'computational thinking' within the construction of indigenous craftwork. These computational concepts and practices form the basis for what we refer to as cultural artifacts in simulation [54]. As we saw in the previous chapter, these computational elements are not always readily apparent, and at times, they take considerable 'detective work'. Still, it can seem like the easy part of the research when the task of reproducing the craft in software is at hand: accurately reflecting the crafting process—both aesthetically as well as in terms of cognitive modeling—while simultaneously being useful to teachers in the classroom, and on top of all that, remaining an attractive activity for students can be enormously difficult and complicated.

This chapter will concentrate on the development of the programmable Culturally Situated Design Tools (pCSDT) for Cornrow Curves, Kente Cloth weaving, and Adinkra stamping simulations. The development of the Kente and Adinkra software applications involved interviews with craftspeople, conducted under a protocol approved by the Rensselaer Institutional Review Board (#998), with the consent of interview participants. These crafts people live in Bonwire and Ntonso, both villages in Ghana. These interviews helped to inform the software development of the applications that followed, using the Agile Software Development method. Once coding was sufficiently complete to deem the software usable, usability testing began at a local junior high school in Albany, New York in order to enable the feedback loop that would inform further development and software 'tuning'. On subsequent visits to Ghana the software simulations were then presented for comments by the Ghanaian craftspeople, in order to ensure that the software was providing respectful and accurate representations.

The pCSDTs—similar to Scratch, Alice, and AgentSheets—are examples of the constructionist genre of computer programs; their goal is to teach both mathematics and computer science concepts to students (and for students without prior knowledge, they

---

Portions of this chapter previously appeared as: W. Babbitt, "An analysis of the programmable Culturally Situated Design Tools from an HCI perspective," presented at the *3rd Annu. Symp., Theory and Research in HCI*, Troy, NY, USA, 2012.

also aim to teach some cultural background as well). Constructionist learning tools grew out of Piaget's constructivist theories of learning, which Papert later refined into logo or "turtle graphics" and eventually the "constructionism" school of thought. Within our constructionist environment, learning begins with the simulated design of a culturally significant visual artifact. This learning process has been referred to as design agency [54], as it can be described by Pickering's 'dance of agency' framework originally proposed as a model of scientific discovery. Here, it is not the professional scientist in dialog with nature, but rather a student in dialog with the software. The student has some goal in mind, but soon encounters 'resistance' in the process as the software fails to perform as she hoped. The student then adjusts the script, which results in further feedback loops of experimentation. Sometimes it is a matter of the student altering her own methods or goals; occasionally one finds a work-around to a barrier as if the software was making a concession. Eventually there is a meeting place between what the software is 'willing' to do and what the student is willing to settle for. The phrase 'trial and error' gives the mistaken impression that there is one right answer and it is just a matter of finding it, but the 'design agency' portrait shows that the process is more emergent than what 'trial and error' suggests. Learning in a constructionist medium is a lot like a population of organisms gradually mutating and changing their own environment in the process of biological adaptation [18].

The original indigenous artifacts, after which we pattern our constructionist-style pCSDTs, have computational thinking deeply embedded in their creation. In many cases, this involves a stepwise process that we can equate to steps in an algorithm. When we look carefully at these cultural artifacts, we find that these stepwise instructions follow in a sequence, with repetition and contingent execution based on the current status of the output. These ideas map directly to the computer science concepts of program flow of control, looping, and conditional statement execution. In simulation, these concepts form the building blocks of constructing cultural artifact representations through computer programming.

The development of software that simulates cultural practices and produces relevant artifacts has involved significant design challenges. As a fundamental starting point, the work of Vygotsky has been crucial to informing our work. His 'zone of

proximal development' (ZPD) describes the user experience that the software must be designed to facilitate. The ZPD represents the point at which the level of difficulty is such that the student is sufficiently challenged in the problem-solving task in which learning occurs, but not so challenged that they become frustrated. Aim too low and the child is bored and does not learn; too high and the child gives up [55].

In addition to Vygotsky's ZPD, Resnick's notion of "low floor and high ceiling" [56], has also been an important software design principle for us. The low floor notion can be thought of as 'low barriers to entry': the software is very easy to start using, offering intuitive and uncomplicated means for a student to start working on a design. The high ceiling denotes a kind of scalability: the software should not limit the student in terms of the complexity of the design they wish to create. Low initial learning curve and 'sky's the limit' design capability offers the student simulation software that can quickly engage a student's creativity, without placing limits on that creativity. The two design principles, together, mean that we seek to keep the user within the ZPD, which constantly changes as they grow in ability.

In terms of the pCSDTs, this resulted in a constant assessment, evaluation, and reassessment cycle in negotiating the software development process to 'get it right'. This cycle took place in weekly development meetings, where the course of development was plotted out for the next week, in light of the previous week's accomplishments. All the while, the primary concern and almost all of our discussions focused on the student user, how much software complexity to show or hide, and how these decisions would impact software use and student learning.

3.2 Software Development

Agile or 'The Agile Method' refers to a set of guidelines, or more accurately, guiding principles that should be used in software development. These guidelines come from the Agile Manifesto and can be summarized as:

Table 3.1: Agile Manifesto [57]

1.	Individuals and interactions over processes and tools.
2.	Working software over comprehensive documentation.
3.	Customer collaboration over contract negotiation.
4.	Responding to change over following a plan.

Agile in the world of 'for profit' software development describes a process in which individuals and interactions emphasize smaller chunks of working code, with constant feedback from customer collaboration, or as it is sometimes called, the "voice of the consumer" represented by one of the design team members. This is far more useful in the design process than obsessing over contract fine print, and it acknowledges the fact that it is difficult to foresee all the consequences of initial design specifications until you have working code in front of you. Rather than prevent 'requirements churn' by locking us into a rigid plan, Agile embraces these new facts as they become known, and it allows rapid ('agile') response to desired changes.

In our world of developing educational software with high pedagogical value, our customers are our student users. We did our best to negotiate design requirements in our weekly meetings, anticipating the user experience and challenging ourselves to extend our abilities as software developers, to write excellent code that would result in excellent applications. In this case, the code did not represent some mundane business application, but instead, it represented an exciting and challenging pedagogical tool that students would enjoy using.

It is likely that Agile is probably the only method flexible enough to handle the nature of our particular labor situation with respect to pCSDT development, given the developer turnover as graduate students join and then leave the effort, let alone the unforeseen software limitations that we discovered only in the middle of developing a particular feature. Agile encourages developer flexibility and good humor by keeping developer focus where it should be, on creating great software.

3.2.1 The Programmable Culturally Situated Design Tools

The development of the pCSDTs offer a rich opportunity to explore software development in the context of multiple and sometimes conflicting goals and user needs; they illustrate the difficulty of 'getting it right' in such circumstances. The pCSDTs offer this opportunity both through concept formulation and concept refinement, where the goals themselves can evolve as revisions reveal new aspects of user interactions. An analysis of the development process is thus relevant to scholars interested in the general

process of software engineering, as well as specific areas such as educational software design and cross-cultural software design. It specifically offers an opportunity to look at software development in relation to the intersection of culture in the artifact to be simulated (both vernacular culture and heritage culture), the cultural identity of the user (which is dynamically constructed as children grow in their understanding of themselves), and the learning content (typically in reference to state or national standards).

It should now be clear why it was necessary to carefully describe the problems of deterministic myths in holding back underrepresented students. If the goal is to make a case for why math and computing is a part of their own heritage—that is, making a case for why mathematical ideas and computational thinking were already present in the culture before European colonists arrived—then the simulation cannot simply create the image of the artifact as seen by an outsider; the simulation must reflect, at some level, the actual process used by the original artisans who create it. At the same time, it cannot be too realistic: if we wanted to exactly recreate the original experience, we would simply provide a pile of beads, thread, or whatever materials were used. As we found, even a purely mouse-driven interface proved to be too concrete to facilitate students' growth in math and computing skills. It must work in a middle ground between symbolic abstraction and physically concrete experience. The pCSDT's purpose is essentially to "translate" indigenous knowledge or vernacular knowledge to a corresponding concept in the classroom, but the means of translation—to extend the linguistic metaphor—must be less like an intimidating list of grammatical rules and more like a friend who coaches you in the new language.

This brings us to the challenge of integrating learning as a goal. In addition to ZPD, another term often associated with Vygotsky is "scaffolding" – like a building created with a temporary series of supports, a moving ZPD should allow the user to build on previous knowledge and skills to get to higher levels. By building on this corpus of "translated" indigenous knowledge, students can feel a sense of continuity with their own heritage culture. This effect is nicely captured by an unsolicited communication our team received from a teacher at a Lakota Nation school using the bead loom CSDT:

> You might be interested to hear that one of the students, who is an IT major, an artist, a very traditional bead worker and fluent Lakota speaker, was so delighted with the software that he decided to go ahead and develop his own algorithms independently. He was really inspired. He said it was the first time that math/graphing seemed to really make sense or 'click' for him. I have not seen how far he got with computer algorithms, but his final project for our math class was full of linear models that described his most recent beadwork creations [58].

It is important to note that the student was using a non-programmable version, which is why the teacher said, "I haven't seen how far he got with computer algorithms." This is precisely why we needed to create programmable versions of CSDTs, so that the inherently algorithmic content of indigenous ideas and practices could be integrated into the learning process from the start, rather than added on later.

The CSDTs began as a set of Flash based tools (that is, they were developed using Adobe Flash). The Flash sets of applications were specifically aimed at teaching mathematics concepts without requiring users to write a program or script that creates their design, as is the case with the pCSDT Java applets that followed them. The Flash versions of the tools looked visually beautiful; they were easy even for very young children; and math teachers were happy with the fit to their curriculum. However, all updates to system variables happen immediately upon the user making changes to numeric fill-in fields in the user interface. It is important to note that this is not leaving the algorithm hidden "behind the scenes"—it is clearly visible to the user (Figure 3.1). While providing the 'low floor' needed, and even exposing them to computational thinking, it limits the user to filling in parameters for algorithms that are already locked into the interface. This pairing of pre-determined algorithms and numeric fill-in fields limits the tool's ability to be useful in fully developing computational thinking in students: they can explore the consequences of an algorithm as they vary its inputs, but not explore its redesign.

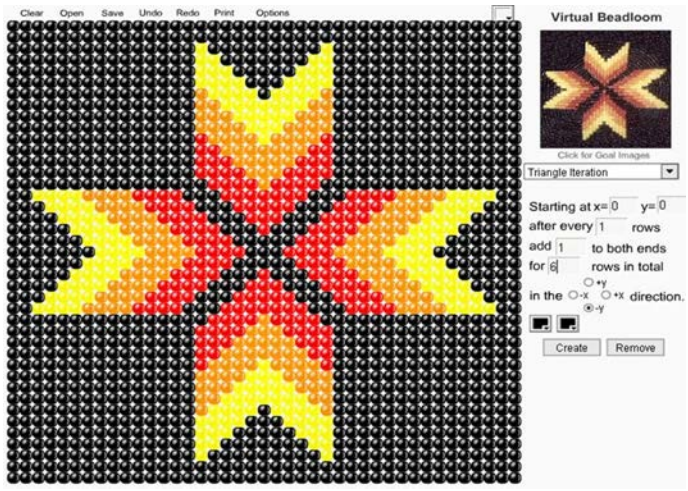


Figure 3.1: The Virtual Beadloom CSDT

The pCSDTs, in contrast, use a scripting panel, similar to those of Scratch, Alice, and other graphical approaches to programming. This scripting panel, shown as the left most column in both Figures 3.2 and 3.3, provides students with the opportunity to iteratively develop solutions to programming problems. In the case of the pCSDTs, this problem or 'design challenge' comes in the form of a design of a culturally significant object or practice. Students drag codelets from the left most column into the middle scripting column, arranging them in an order that they think will accomplish their design goals. The student then checks the results of their actions by clicking the 'run' button. The result of their work in the scripting panel will show in the output panel on the right of the interface. If the design in the output panel matches the student's expectations, the student has not learned anything new, but has instead successfully used the knowledge he or she already possessed. It is when the output does not match the student's expectations that the magic of learning takes place.

When the student clicks the 'run' button and has an unexpected result appear in the output window, the student then begins to think 'What did I do wrong?' or 'How do I fix it?' or 'That's not what I wanted.' As the cognitive wheels turn and the student reviews the sequence of codelets that they have attached in the scripting panel, computational thinking takes place. Perhaps, their thinking is not completely taking into account all of the nuances of the changes that their codelets are creating in the software system. As time passes and the student mentally works through the script, step by step,

connections begin to form between objects and then relationships appear. When given sufficient time and effort to form these relationships, learning takes place, conceptual changes occur, mental models [59] are updated, and the desired design outcome takes shape.

3.2.2 The pCSDT Architecture

The current iteration of the pCSDT program, called CSnap, is written in JavaScript and is shown in Figure 3.2. However, the bulk of my design work has been with the prior version of the pCSDT applications. This version shown in Figure 3.3 was written in the Java programming language and was deployed on the CSDT website using Java applet technology. Building the pCSDTs as Java applets allowed us to code a complex application that could be streamed over the Internet to a user's computer, without the application needing to be installed on the local machine.

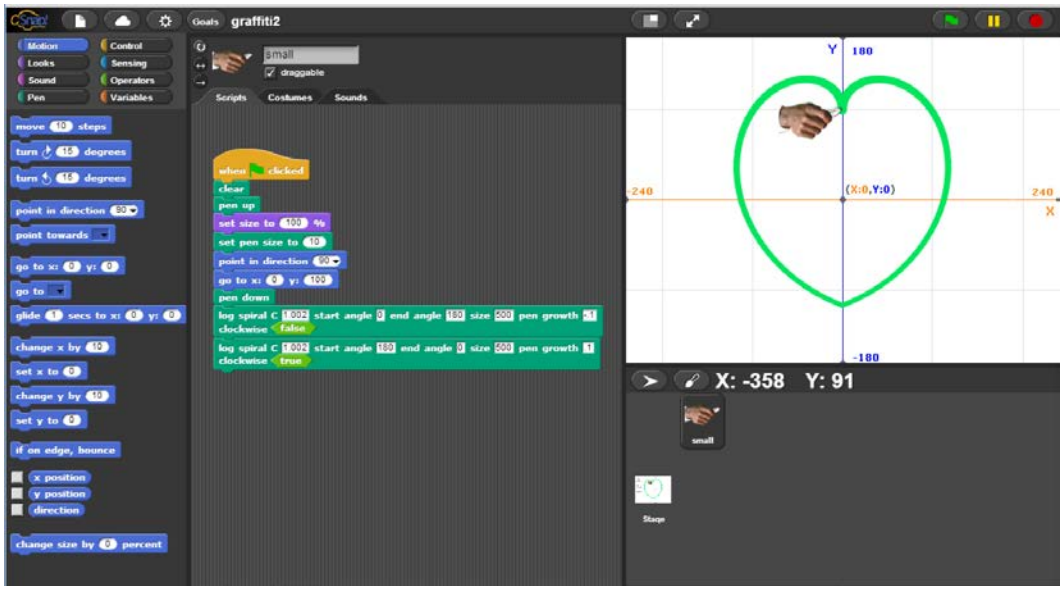


Figure 3.2: CSnap interface with the Akoma project loaded

The Java applet programs were designed with a 'Core' Java class, containing the code for the features that were present in all of the pCSDT applications. This base class was responsible for the graphical user interface GUI, which included all of the graphical attributes such as the application window, panels, menus, buttons, and text boxes for data

input. The Core class also contained the code for all of the backend processing needed for handling the scripting features of the application. This modular design, with functionality common to all of the applications placed in the Core Java classes, was then extended using Java Reflection, allowing application classes to extend the Core class.

The pCSDT applications, such as the Cornrow Curves, Kente Cloth, and Adinkra stamping simulations, all had their application specific code in a set of classes that made use of the Core class, through Java Reflection. Utilizing the reflection @Override annotation, the code in the application method where the @Override was present then executed instead of the Core method definitions with the same signatures. This process allows for functionality in the application to be added dynamically at run time, even though it may not be known at compile time.

The interactivity of the GUI is handled through a 'game loop' that cycled through application data structures, recognizing user input and changes to system variables in a predictable fashion. Once the GUI has started and drawn to the user's screen, the software enters into the game loop, constantly checking for changes made by the user. The loop would start with the Starting Values properties pane to pick up changes to system object attributes, such as screen output size, Cartesian grid color, and the like. Once changes in system variables were recognized, the game loop would proceed to the object queue, which served as a list of objects currently instantiated in the application, all needing to have initial values and programmatic changes recognized. The game loop then worked its way through the object queue, item by item, recognizing changes made by scripts using codelets in each object's event queue. Once the game loop completes a cycle, it repeats until the user closes the user interface, exiting the program.

3.2.3 The pCSDT User Interface

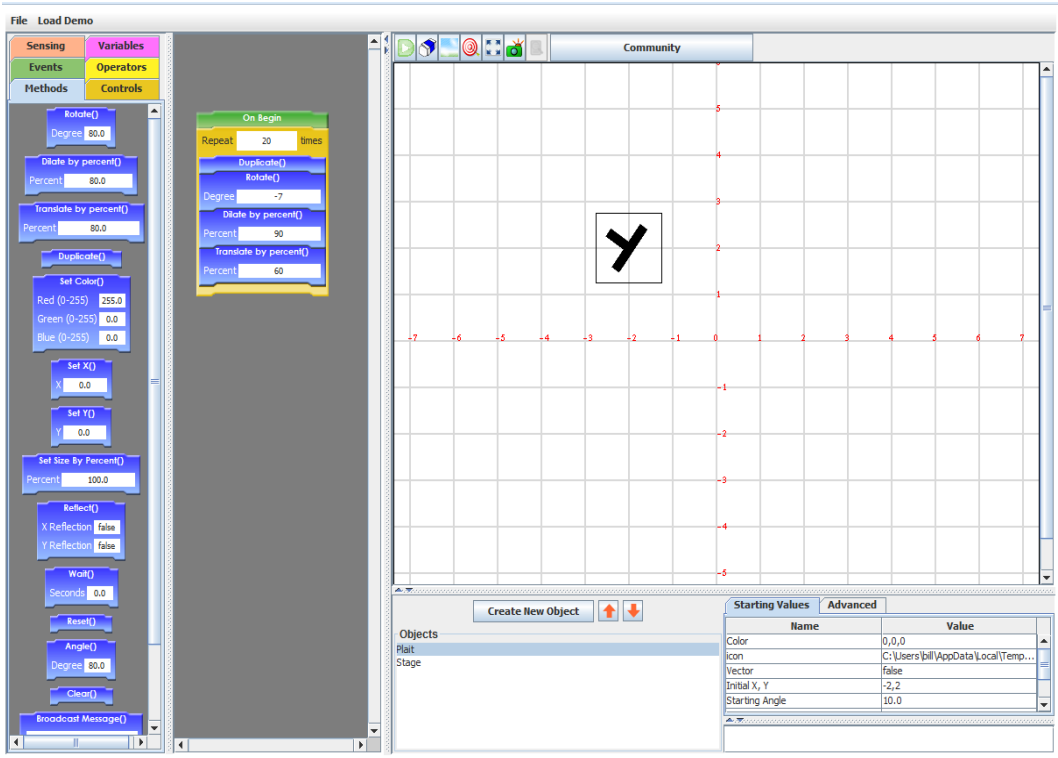


Figure 3.3: The Cornrow Curves simulation

When the pCSDT user interface opens on the computer screen (Figure 3.3), it reveals a series of panels organized from left to right. The left most panel contains tabbed panes that contain the building blocks that student users will use to construct their projects. Each tab in the panel contains building blocks called codelets that are grouped by function. Placing the codelets in one panel, grouped by function, maps directly to the 'toolbox metaphor' as is used in the human computer interaction literature [60].

Immediately to the right of the codelet panel is the scripting panel, where the codelets are assembled into small programs. The output window is located to the right of the scripting panel and occupies the majority of the user interface. The output panel is where all the action happens as the user created script is executed. In addition, there are two smaller panels underneath the output panel, which list the objects that have been created in the interface (lower left), and their initial values (lower right).

The codelet panel has tabbed panes that are color-coded and labeled 'Event', 'Method', 'Controls', 'Operators', 'Variables', and 'Sensing' (Figure 3.3). Each panel

contains codelets, the same color as the label on the tab pane that performs particular functions. The group of codelets, within each pane, perform similar functions within the scripting system. Similar to a toolbox, a user accesses the appropriate panel, drags the desired codelets into the scripting panel, and attaches the codelet to those already in the panel. By assembling the codelet building blocks in a particular order, the user writes a small program. When the user clicks the 'run' button, the result of the script will be displayed on the output panel.

The green Events pane contains codelets designed to respond to user interface events. For example, the Events panel has a codelet entitled 'On Begin' that responds to the User Interface (UI) event of clicking the 'run' button. Once a user clicks on the 'run' button in the UI, this triggers the event queue, which systematically works its way through all of the objects currently created in the system. For each object, the system finds the event 'On Begin' and sequentially executes all of the codelets attached to it. This systematic path through the event queue creates a loop, updates the attributes of each object, and displays these updates in the output panel.

The blue Methods panel contains codelets designed to set the attributes programmatically for each object currently created in the system. The method codelets expose these attributes to the user, allowing those attributes to be altered during script execution. As the event queue is processed, following the click of the 'run' button, and as each 'On Begin' codelet is located for each object, the methods attached to the 'On Begin' codelet are executed in sequence. This execution updates the system values for each attribute in the codelet, such as color values or Cartesian plane coordinates. These changes are then reflected in the update to the output window.

The orange Controls panel contains codelets that allow the user to incorporate traditional programming concepts, such as looping for a fixed number of times, forever, or while a condition holds true, as well as conditional execution. These control codelets allow the user to alter the sequence or 'flow of control' within the script execution. Changing the flow of program execution in the scripting panel is very similar to what happens in a traditional programming language.

The yellow Operators panel contains codelets that allow the student to use relational operators to create condition statements for the event stack to test for

conditional script execution during the execution loop. These relational statements can incorporate system variables such as object attributes listed in the orange Sensing panel or user-defined variables created in the pink Variables panel. The Sensing panel lists object attributes that have their initial values set through the Starting Values panel, and those initial starting values are programmatically changed during script execution by the method codelets.

In addition to the panels described above, there is also a button bar above the output window. The button bar contains buttons that users click to perform certain actions. The most important action from the point of view of the user is the 'run' button. Clicking the 'run' button toggles the button to 'stop' and sets the system Boolean values, such that the game loop begins updating the interface by including the object queue in its cycle through the system objects. A second click of the 'stop' button sets the toggle back to 'run' and the game loop stops updating the interface by excluding the object queue in its cycle. Next to the 'run/stop' button is a button for erasing the output image, loading a background image, making the goal image panel visible, making the user interface go full screen, and lastly, saving the output window as a graphics file.

3.2.4 Cornrow Curves Simulation

The Cornrow Curves pCSDT application followed a very popular, robust, and well thought out CSDT that simulated the cornrows hairstyle. The initial work of translating between the knowledge of braiders and the simulation had already been done before I arrived. As such, it was a straightforward task to take the already completed design work from the CSDT and simply convert it into a pCSDT. Figure 3.4 shows an early version of the Cornrow Curves pCSDT. The interesting thing about this picture is that it offers us a glimpse at what the early pCSDT interface looked like. As noted above, the Cornrows application code was written in Java classes that sat on top the Core classes. Therefore, with little exception, what is shown in Figure 3.4 is the work of the Core classes. Consider that what the Cornrows Java classes contributed were the specific method codelets in blue, and the graphic 'Y' that is the plait image.

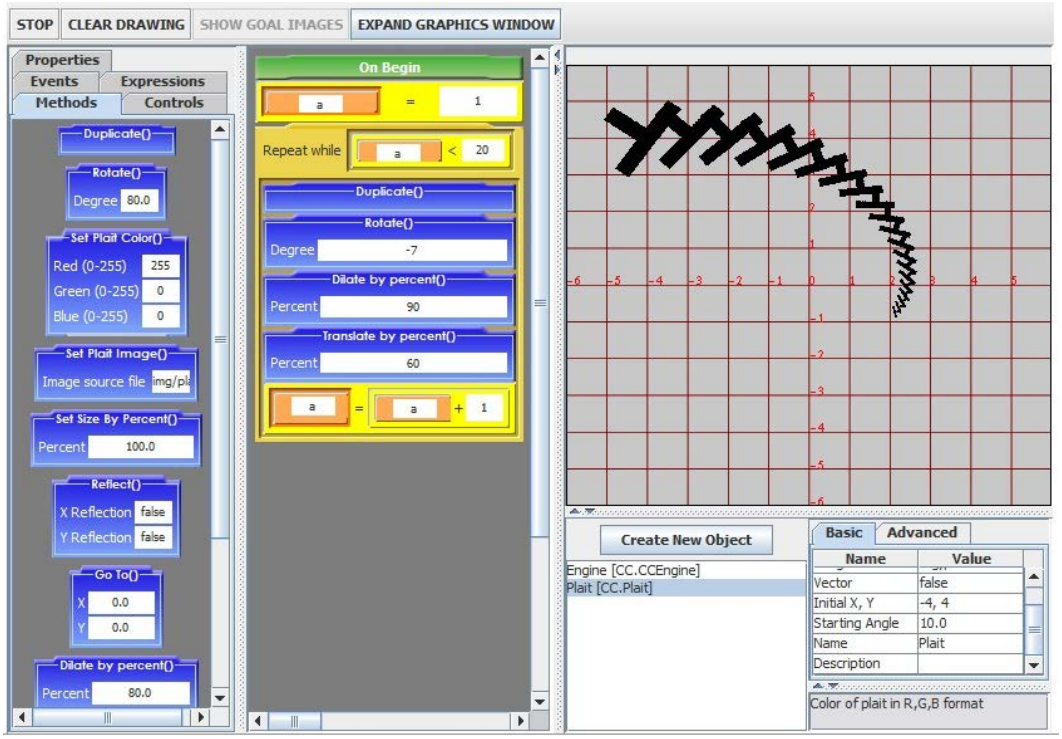


Figure 3.4: Early version of Cornrow Curves simulation

My work on the Cornrows applet involved solving software issues that were occasionally made visible through usability testing. In addition, I worked on converting the drawing methods to allow for animation of the plaits. Maintaining the Cornrows' code often meant making minor (and major) adjustments to the application code, as changes and upgrades were made to the Core code. The interesting thing about working on Cornrow Curves was that it was the first pCSDT applets that I felt I really knew comprehensively.

Each of the pCSDTs had their own idiosyncrasies and Cornrows was no different. For example, the first plait was in a data structure all by itself, and so changes to the methods that affected the appearance of subsequent plaits needed to be applied to the first plait. Likewise, calls to the reset method that occurred when the user clicked 'clear' needed to be applied only to the first plait, because all of the other copies of that image that made up the braid were deleted from the data structure that contained them.

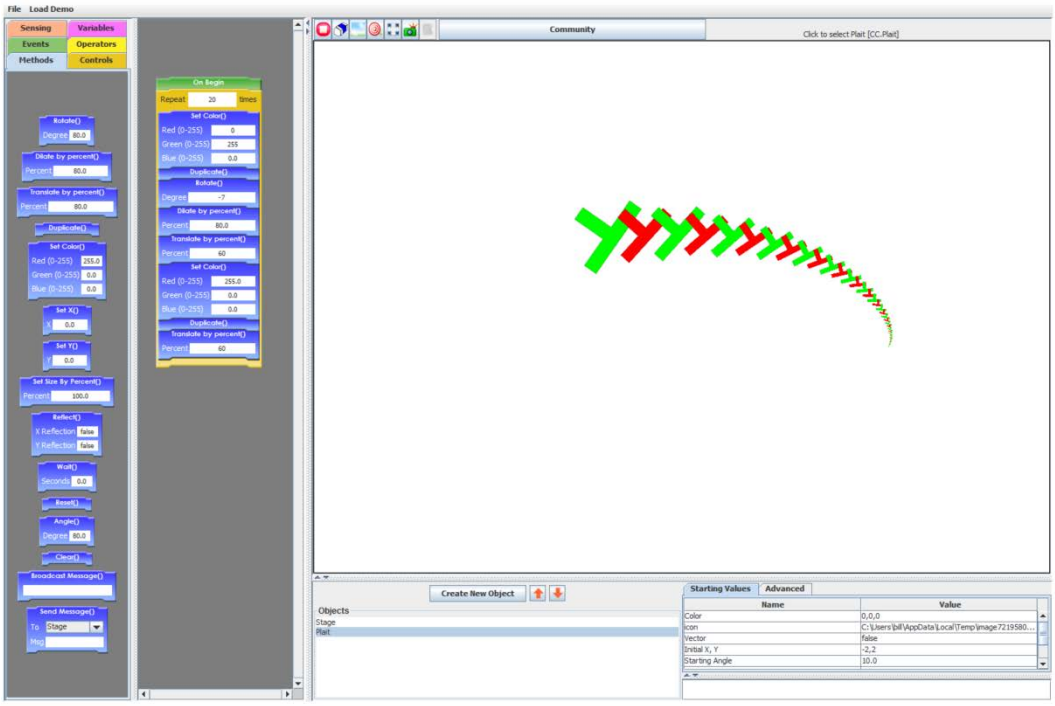


Figure 3.5: Final version of the Cornrow Curves simulation

Animating the braid was an interesting challenge, seeing that the OpenGL output window wrote all changes to the screen at once, which meant that the whole braid would just simply appear in one pass of the game loop. Creating the appearance of animation involved indexing the plait copies with a variable that determined when the plait should 'appear' on screen, and then saving the copies in a data structure for when they were needed. The data structure for the screen would be populated with plait objects at the beginning of a game loop, calculated all at once for when to appear, how long to appear, and at what location. Then, as the game loop cycled, plaits would be displayed when and where (as already calculated), based on the number of the game loop. Like actors waiting their turn off stage, when the appropriate game loop occurred, they were summoned on stage to play their part.

The Cornrows applet has an interesting feature that not all of the pCSDTs offer. Cornrows allows the user to import a graphics file to replace the plait image in the simulation. This ability allows for 'braids' using, amongst other things, the faces of friends, sports equipment, pets, or other familiar images. There is a tension at work here: on the one hand, simply providing a blank slate and allowing other artifacts to be loaded

into the software and used runs the risk of being overrun by corporately created media, gun violence, and a whole host of undesirable content. On the other hand, forcing students to remain within a narrow domain we define can restrict their creativity. The goal is to find a happy medium where we offer positive directions to begin with, but allow them the freedom to take their own path after that. These issues are discussed at length in chapter 4 of the work.

3.2.5 Kente Cloth Simulation

I had the good fortune for being involved with the Kente Cloth pCSDT from the very beginning, when we visited Ghana in the summer of 2011. Unlike the cornrows simulation, I was involved with the initial research to determine what math and computing ideas could be gleaned from discussions with the indigenous designers. The design considerations for a Kente Cloth simulation pCSDT were greatly informed by the time spent talking with Richard Bonsu (see Ethnographic Case Study 1 in 3.4.1). As is clear from the first attempt at a Kente simulation in Figure 3.6, the image in the output window does not do justice to the fabric showing in the Goal Images panel (same figure, upper left). Richard provided me with a complete understanding of how he went about his craft, including explaining the different patterns of Kente that he weaves, and he helped me to develop an appreciation for the time commitment that each creation requires.

In working through the design considerations, it seemed clear that the simulation should be arranged so that the output window would contain only the first quadrant of the Cartesian plane. Students would locate their blocks of color in their virtual weaving using only positive numbers for the X and Y coordinates. However, when we consider the complexity of this tool, it seems that the majority of the work performed by students is limited to reading the Cartesian plane coordinates and then iteratively creating blocks of color in the output window using loops and other control codelets.

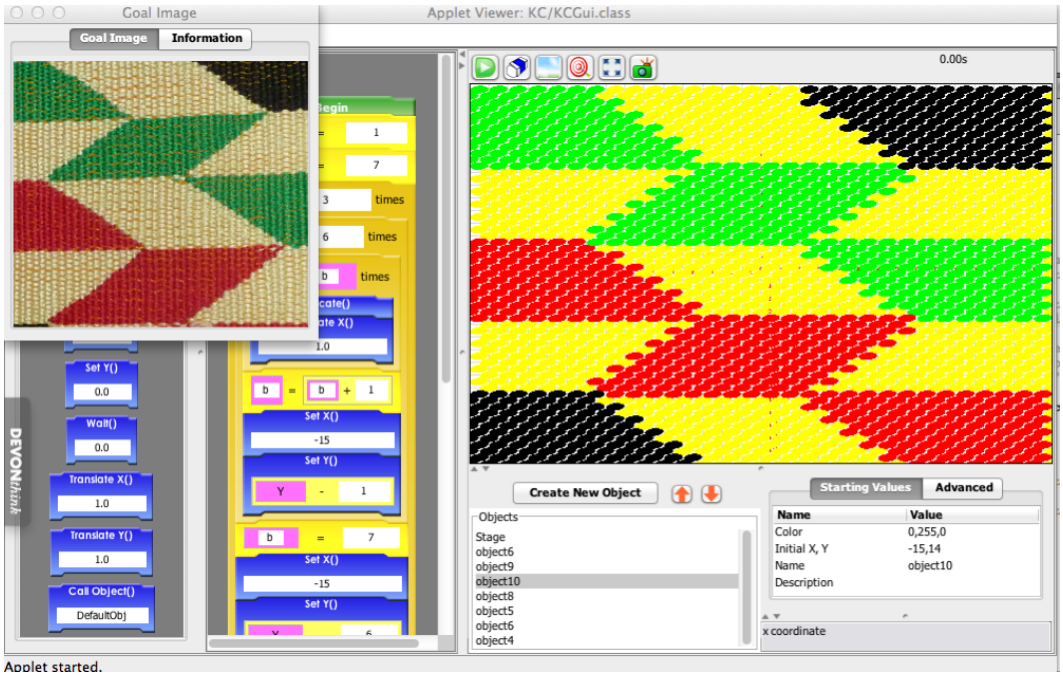


Figure 3.6: Early version of the Kente Cloth simulation

In Figure 3.7 there is a screen shot of an earlier version of the Kente Cloth weaving simulation software. This version used Java2d circles to render the 'weave' pattern in the output window. As it turned out, this was a huge software engineering mistake because although it worked with a sufficiently robust modern laptop computer, there were far too many objects for a netbook computer of only moderate capabilities to handle. This version of the software turned out to be incredibly slow and a big disappointment to our student software testers at the junior high school in Ghana (see sections 5.3.2 and 5.4.1 for more on the Kente Cloth simulation).

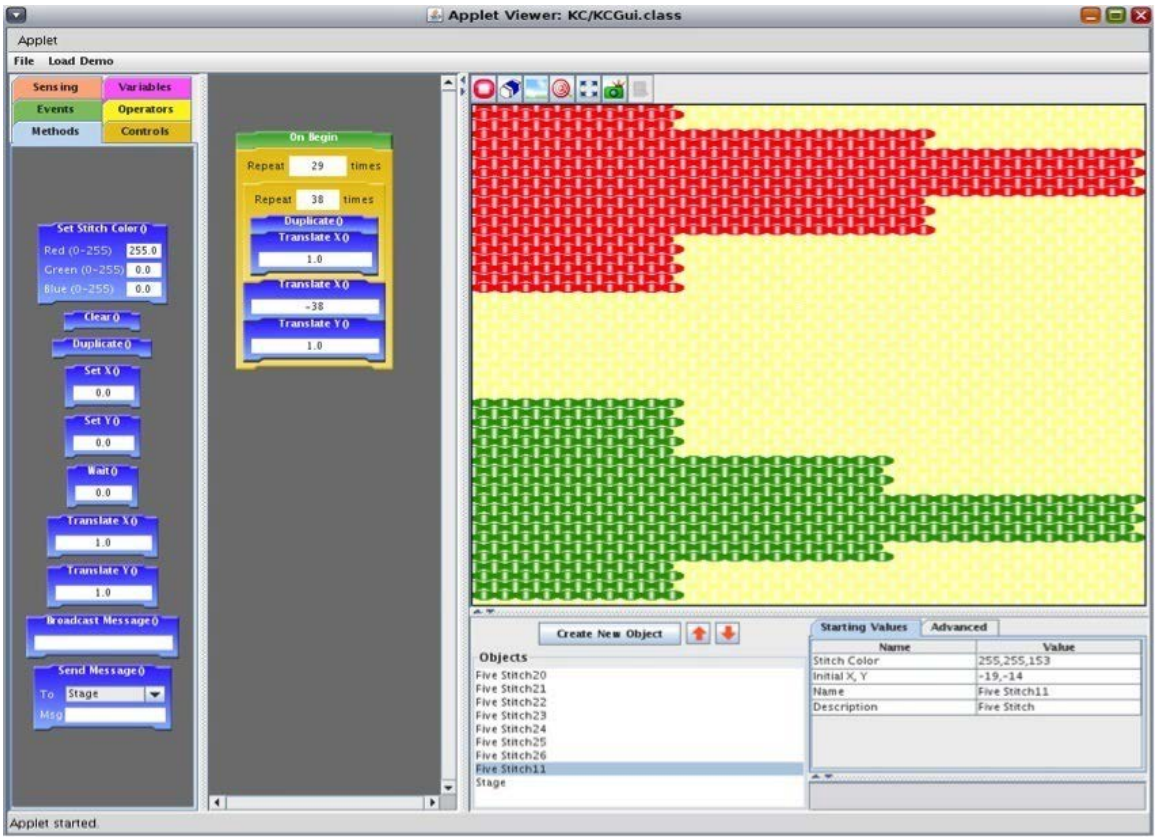


Figure 3.7: The final version of the Kente Cloth simulation

The Kente Cloth application shown in Figure 3.8 is the final version of this software. The output image rendering uses OpenGL ellipses to simulate the vertical and horizontal aspects of the 'weave'. This version of the software was much faster than the original Kente Cloth version and was considered a success with students at the junior high school in Ghana where the software was tested in the summer of 2013.

3.2.6 Adinkra Simulation

I also had the great fortune of being involved with the Adinkra stamping pCSDT from the very beginning, interviewing master textile artisan Gabriel Boakye initially in 2011, and then presenting him with our version of the software in 2012. The interview with Gabriel was very similar to the one that I conducted with Richard in Bonwire for the Kente Cloth simulation (see 3.2.5 of this work). Gabriel was very generous with his time, showing us the ink making process, how stamps are carved, and even showing us how to stamp our own Adinkra cloth patterns.

I was able to determine from the interviews with Gabriel that the Adinkra stamping simulation would be more valuable as a pedagogic tool if we focused on the geometry present in the Adinkra symbol shapes. Many Adinkra symbols feature representations of a logarithmic spiral, as was pointed out long ago in professor Eglash's *African Fractals* text, and present a perfect opportunity to leverage the mathematics involved as a form of cultural capital for African American students in the United States. The simulation of such symbols would be a challenge, as stated previously we would need to hit the "sweet spot" of fidelity to the craft representation, mathematics at the appropriate grade level of difficulty, and keeping the scripting difficulty within Vygotsky's ZPD.

Having the big question answered concerning the primary focus of the simulation, there were still many little questions that came out of our initial interview with Gabriel. Because of the mathematical challenges inherent in simulating shapes such as logarithmic spirals, it seemed that also having to negotiate all four quadrants of the Cartesian plane would be a nuisance to students. Thus, it seemed clear that all the action of the simulation should take place in the first quadrant. There were still unanswered questions. We wondered if we should animate the appearance of the Adinkra symbol in the output window, or have it appear all at once, as it would if it were an Adinkra stamp. It seemed to us, that in order to maintain fidelity to the practiced craft, having it appear all at once would be best. However, in the end, we decided that animating the appearance of the Adinkra symbol would be a lot more interesting to students.

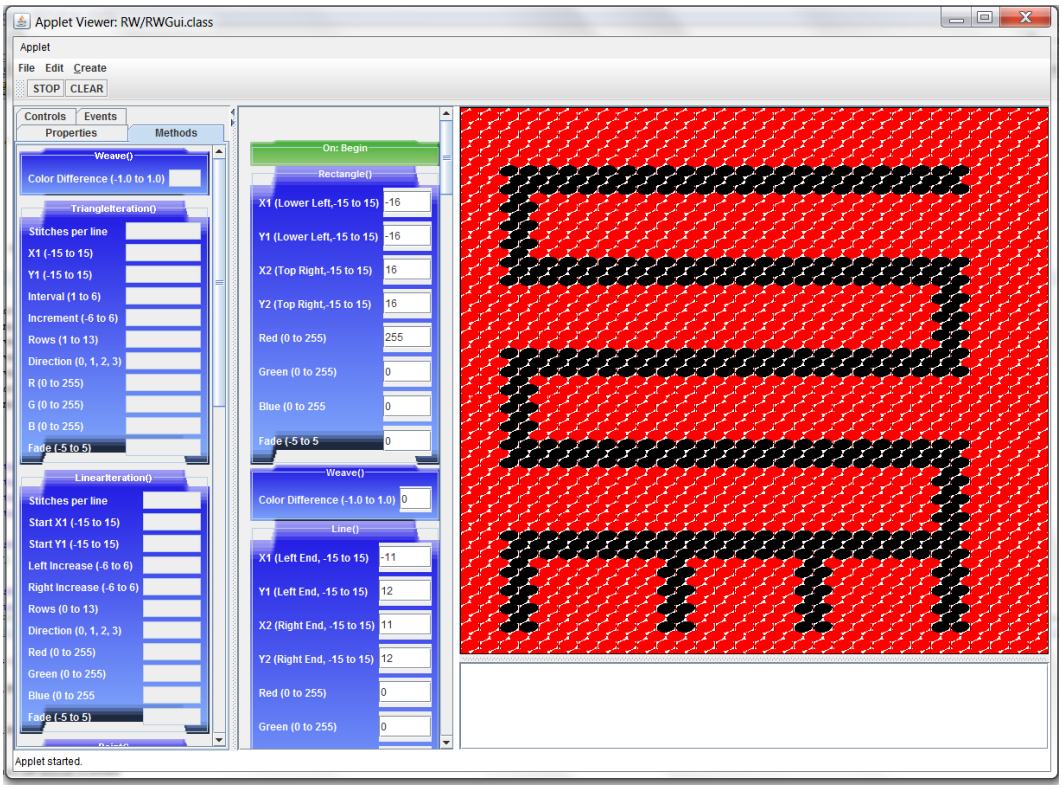


Figure 3.8: Early version of the Adinkra stamping simulation

Figure 3.9 is a screen shot of an earlier version of the Adinkra stamping simulation software. This version used Java2d circles to render the stamp pattern in the output window. Thankfully, this version of the software never actually beta tested with student users. If it had been, it would have certainly overwhelmed the capacity of moderately powered netbooks to calculate all of the circles that made up the image (see 5.4.1 for the effect of Java 2d circles). In the end, this version was scrapped because it did not properly render the image in the same way that an Adinkra craftsperson would have produced one, as the output image looks much more like a weave than a stamp.

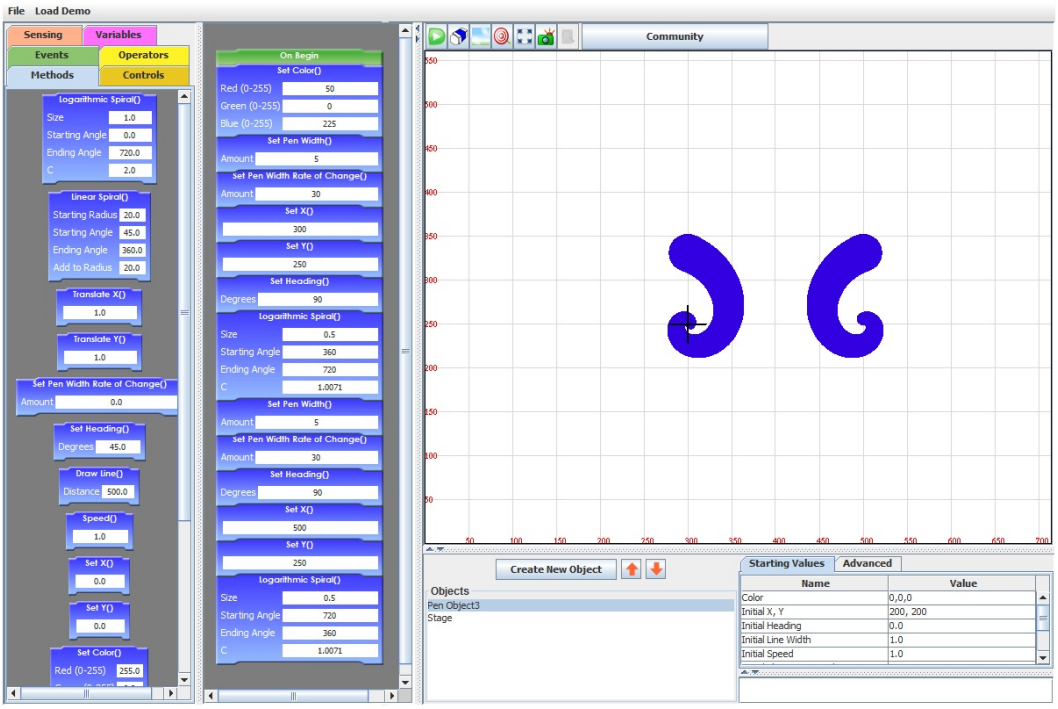


Figure 3.9: The final version of the Adinkra stamping simulation

Figure 3.10 shows the final version of the Adinkra stamping pCSDT applet that we deployed on the CSDT website. We follow the development of this pCSDT using ethnographic developer stories in section 5.4.2 of this work. In addition, we report on an experiment conducted using the successor to this pCSDT, called CSnap, simulating Adinkra in chapter 6.

3.3 Human Computer Interaction and the pCSDTs

In addition to the software development effort for the pCSDTs, this work has involved a great deal of human computer interaction research. The HCI aspect of this research has focused primarily on user stories, as their experiences shape both the needed debugging of the software as well as the programming direction. This exploratory use of the pCSDTs, at the time considered in beta release, helped to quickly point out difficulties in the software design that were able to be addressed much sooner than would have been possible otherwise. From an HCI perspective, it is especially important that we consider user mental models and cognitive loading.

3.3.1 Mental Models

Mental models are "conceptual organizations of information in memory [61]", that allow us to develop additional inferences about what we understand. [62] Mental models are not new, and have been studied and discussed quite extensively over the years. The understanding of the mental model representation is to help us imagine how we apply previously learned concepts to new, unfamiliar concepts, which result in extending our understanding. When we consider how users react while using our software, mental models are extremely important because they help to inform us of the user's frame of reference.

The mental model that most computer users bring to their interaction with computer software is that of the desktop. All operating systems offer some analog of the physical desktop, from graphical icons of stacks of documents, to the trash bin where objects are removed from the system. The desktop metaphor for operating systems allow users to instantly feel at home when using their computers, allowing them some intuition about how system objects on the screen can be expected function, similar to those in the real world.

When the pCSDT tools open, the user is immediately greeted with small objects that look like puzzle pieces. These puzzle pieces provide the user with some intuition that they snap together, to build larger objects, in this case small programs. Where most graphical user interfaces in operating systems have their design based on the desktop metaphor, the pCSDTs have as their design base the 'Toolbox' metaphor. They also bring to the tool a preconceived notion about how tools fit in a toolbox and how they can be organized by function.

The desired outcome for the pCSDT user is to learn new concepts in mathematics and computer science. This learning amounts to the extension of their existing mental model to incorporate the tools that they used to construct their onscreen artifact. A successful outcome will be one in which the user extends their mental model of the initial toolbox of the user interface to include the new tools used in building the onscreen output. Based on student interaction with the 'Repeat While' control codelet, for example, they will have developed a mental model thought process similar to, "If I construct a condition, the codelets placed inside the 'Repeat While' will execute only

while that condition remains true, and when that condition becomes false, the changes being executed will stop. I will observe this start and stop on screen in the output window." The proper graphical feedback then is crucial. It is based on this feedback that the student will succeed in validating their new extended mental model of how these programming concepts work. If the UI responds in an unexpected or incorrect way, the user will fail to validate their new mental model and will not learn the new concepts.

The student's success or failure in learning programming concepts (e.g., the use of control structures, variables) requires that the designers of the user interface walk a very fine line between design that outright instructs and one that allows for user exploration. This is the creative tension that we, as developers, need to be careful to balance in this type of educational software. If the tool is too difficult, the student will become frustrated and give up. If the tool is too easy, then the student will become bored and will miss potential learning opportunities. An additional strategy in this UI tuning is to find ways to reduce the cognitive load that the users face from other interface elements.

Mental models also play a large part in how crafts people understand and practice their crafts. The interviews with Gabriel and Richard (sections 3.2.5 and 3.2.6) revealed to me, how very different Kente Cloth weaving and Adinkra stamping are from each other. The difference between the complexity of Richard's mental model concerning Kente Cloth and Gabriel's mental model of Adinkra seemed very striking to me. They were both processes, but the orders of magnitude of complexity involved seem to make them 'different' kinds of processes. Richard needed to hold lots of information in his head for the different types of weave patterns he regularly created. Gabriel, on the other hand, it seemed, did not need to maintain nearly as much information in his mental model for his Adinkra craft.

3.3.2 Cognitive Loading

As stated previously, the developer of the tool needs to walk the fine line between concealing the complexity of the underlying software behind codelets that do more, at the risk of reducing learning opportunities for the user and possibly making the tool boring to users. The alternative is to expose more complexity in the underlying software by

designing codelets that do less, at the risk of increasing user frustration with a design tool that has become much more complicated in its use. In addition to the level of cognitive complexity chosen for the codelets, there are additional ways to reduce cognitive complexity to support user learning.

In the considerations that have been discussed throughout our design efforts for the pCSDTs, the most important has been to support the user in whatever way we can, without diminishing the opportunity for student learning. The user interface design considerations that have helped in this support has been the selective display of information, in particular, the availability of codelets through the use of panels, and by further limiting the display of codelets to those belonging to the currently selected object. This limitation of codelets reduces the cognitive load for the student, which makes available to them only the current relevant codelets from which they have to choose in constructing their program script.

When the UI starts, it loads a default set of objects and the scripts created for that set of objects. The scripts use codelets that are from the Methods panel for each object. At this point, some small demonstration program is available to run, getting the users 'feet wet' in how the tool works.

From here, the user can then customize the default script, extending its functionality and thus extending the output results in a desirable fashion in the output pane. With the UI starting with an active object selected, the method codelets available to use for that object, and a small script made from those codelets, the universe of possible scripts has been limited to a much more readily understood starting point. Color-coding for control, method, and event codelets easily allows the user to identify which panel will contain additional similar codelets for use in extending the script further.

As the user gains familiarity with the scripting panel and the already displayed codelets, the time will come to expand the creation, and the student will create additional objects. Each object is selectable through the object listing, and upon selection, the relevant codelets will populate the method pane for use in that objects scripting panel. The user has already gained familiarity with the method codelets and scripting panel of the first object, therefore starting work on another object should feel comfortable to the user. This familiarity reduces the cognitive challenge to simply understanding the new

codelets presented that, perhaps, were not available in the first object. In addition, familiarity at this level will probably encourage the student to begin exploring the control and operator codelets to create additional, more interesting designs. This progressive disclosure of codelets helps to reduce cognitive load by limiting the interface elements that are visible to the user at any one time, to only those elements that are relevant to the task capable of being performed [63].

Another way the UI assists users through reducing cognitive load is to provide on screen scaffolding in the form of (possibly) familiar tools such as Cartesian grids for object placement in the output screen. When the pCSDT is not in the 'running' mode, a Cartesian grid appears over the output, and all objects are returned to their initial positions. This grid then allows users to more easily see where in the plane their object should be placed, based on the ordered pair location possible within the Cartesian grid. When the tool is in 'running' mode, the grid is no longer drawn, and thus, it does not appear in the final output of the artifact.

An interesting way that the UI can be used to create motivation in a student is to provide goal images. True to reducing cognitive load, these images need to be purposely selected before they appear in their own separate window, but once they are available, they provide a valuable resource for the student as they create their own design. Sometimes a student does not have sufficient motivation to create a design entirely on his or her own. However, a great deal can be learned through reproducing a design. In addition to the reproduction of the design, there can also be the challenge of reproducing a design efficiently, or more efficiently than a competitor. Team design can result in a deeper understanding of the tool than just a single person struggling through one's own efforts.

Using tutorials is another way that the UI can be used to assist in the reduction of stress in the cognitive load of learning a new tool. Tutorials can provide small toy scripts that behave in typical ways to get the user started in building their own designs, and they can be used to communicate both basic functions as well as more complicated scripting techniques. Tutorials can be especially helpful when the tool has exposed more of the complexity of the underlying software to the user, thus helping to build familiarity with

the program. Seeing how those codelets function within a tutorial can reduce user stress and help users learn the codelets abilities more quickly and effectively.

3.4 Ethnographic Studies

3.4.1 Case Study: Adinkra Stamping

Gabriel Boakye, Ntonso, Ghana 2011.

On a visit in 2011, to Ntonso, a small village in Ghana, I met with Gabriel Boakye, an Adinkra craftsperson of many years. Gabriel has his own shop by the side of the road that runs through his town, which he and his extended family operate, selling their Adinkra stamped fabrics to tourists. We were there as part of a research group from RPI, and Gabriel taught the entire group about his craft. He began with the process of making the Adinkra ink, which he does with the help of his family.

Adinkra ink is made from the bark of the Badie tree that grows in the north of Ghana. The bark is shipped to him from the north, and he processes it into ink. First, he trims the outer bark off the bark pieces using an enormous machete. Once that is complete, the bark is put into a giant mortar and beaten with a pestle until it is broken up. This part of the process is a lot of work; the pestle is heavy and needs to be lifted overhead and slammed down on the bark repeatedly. Once the bark has been broken up, it then looks more like a red fibrous material than bark. It is placed into a large barrel that is filled to the top with water so that the fibers can soak. After a couple of days of soaking, the water takes on a thin reddish color. At this point, Gabriel cooks the bark water down, which gradually thickens the liquid, and as it thickens, it turns black. The bark water cooks over an open fire in close proximity to the families' living quarters. It is easy to see that their source of income could easily have detrimental effects on the families' health.

Gabriel's brother, Paul, is in charge of carving the Adinkra stamps. Gabriel tells us that each of the stamps has its own unique meaning, and that people combine different stamps on a cloth to tell a story or convey a message. Adinkra has its origins in the funerary arts, with the idea that if you stamp a message on a cloth, the departed family member will see it and know what you are trying to say to them. The Adinkra stamps are

carved out of the calabash gourde. Paul has carved many Adinkra stamps in his lifetime, and on the day I visited, he had close to a hundred available for sale, all arrayed in neat rows on a table next to the hanging Adinkra cloth that is, likewise, available for sale.

We each bought Adinkra stamps from Paul, brought them over to the large table where we were able to make our own Adinkra cloths using the stamps we just purchased and strips of fabric that were available in piles on a table. Gabriel and his family make far more money on the Adinkra cloth that has been pre-stamped, but if there is enough foot traffic, the 'stamp your own' business is nearly as good. The entire operation is really a family affair, where younger members of the family benefit from selling the blank pieces of fabric for use in the 'do it yourself' stamping area. Older family members each have their section of the hanging display areas for the pre-stamped products.

3.4.2 Case Study: Kente Cloth Simulation

Richard Bonsu, Bonwire, Ghana 2011.

On a visit in 2011, to Bonwire, a small village in Ghana, I met with Richard Bonsu, a Kente Cloth weaver with many years of experience. Richard works in a Kente Cloth cooperative weaving and retail shop just off the main street that passes through Bonwire. The local economy was not doing as robust as had been in the past, as there were far fewer European tourists passing through town than usual. Richard suspected that this was due to the current state of the economy. However, he wasn't completely certain, though he did know his income was down, and his cost for supplies (thread and such) were going up. As I talked with Richard, it became clear to me that because of the lack of customer traffic, he had plenty of time to spend with me, and I greatly appreciated his willingness to do so.

Richard and I talked at length about how he did his weaving, sitting at a loom that had the warp threads extend approximately 20 feet from where he sat. He informed me that the pieces of Kente cloth that I saw hanging from the walls around me, were all composed of individual narrow strips, sewn together to make larger pieces of cloth. The length of the individual strips ran between 3 and 5 feet depending on the size of the cloth that he had in mind to make. Each strip would be bound off when it was completed and then it would be cut to match the length of several other strips in preparation for the

sewing together to make the larger fabric. As he worked, he rolled up the completed strips of fabric on a spindle as he continued to weave the next strip.

I asked Richard how he decided what design to make. Richard responded that it was important to maintain a selection of different designs, and he knew what would sell from his years of experience. He went on to tell me that each design has a name, he was currently working on "Your Hearts Desire," which seemed to incorporate geometric figures in purple, pink, and blue. Other patterns were called "No One Man Can Rule a Country", "Education is a Better Choice", and "Fathia Nkrumah". The "Fathia Nkrumah" was a beautiful design done in black, green, and gold threads in honor of the first Ghanaian First Lady, Fathia Nkrumah.

I asked Richard how he managed to keep track of where he was in the pattern on which he happened to be working. Richard responded that it was very easy for him by stating, "I keep it in my head". I pressed, "Really, but it seems so complicated, and the strips come out so exact when they are seen sewn together, you wouldn't know that the strips are separate pieces of cloth." Richard responded, "Yes, I know, I have been weaving for many years, and often the same pattern, so I just know, I have it in my head." I asked him if he counted how many stitches or used anything to help gauge the size of the pattern. He said no, he used to when he first started many years ago, and he might count now if he got distracted but really, "he has it in his head."

This first encounter with Richard was very interesting to me. As a Kente Cloth weaver, he created many different patterns, and clearly, he was very skilled at his craft. However 'he had it all in his head', according to our conversation, which seemed difficult for me to believe. I knew this would immediately be the difficult part in building a Kente Cloth weaving simulation. How would I ever manage to capture the particulars of the weaving simulation, if it was all 'just in Richard's head'?

Thinking about this for a long time following our first meeting, I began to grasp, even if in just a glimpse, the model of the craft that Richard maintained all in his head. This mental model of weaving, which clearly had made the craft second nature to him, would not be suitable for simulation. The challenge of accurately recreating Richard's mental model of weaving would be enormously complicated and would not likely yield a

pedagogical tool that would be useful to students. I needed instead, to concentrate on just the physically observable process itself.

Richard Bonsu, Bonwire, Ghana, 2012

I met up with Richard in the same weaving cooperative as I had the previous year. He immediately recognized me, smiling and calling me 'Professor'. As we talked, it seemed that he had plenty of time to share, and I was once again very grateful for his willingness to spend his time talking with me. I opened my laptop and showed him the Kente Cloth software on which I had been working. He seemed genuinely astonished that anyone would have an interest in creating a computer simulation of his Kente weaving.

I demonstrated the software, dragging codelets into the scripting panel and clicked 'run', which created a color block of 'weaving' in the output window. I immediately knew that he thought something was wrong by the expression on his face. Still very happy that someone thought his weaving was important enough to simulate on a computer, but ever so reluctantly, he stated, "That's not how my weaving works, Professor." "My weaving starts from the bottom and goes up the threads, that weaving (pointing to my screen) shows wherever you put it." I replied, with "Ah, yes you are right. We talked a lot about how you weave. We decided that it would be very difficult to show the weaving the way you do it – so difficult that we were afraid that students might give up and stop using the program." "Yes, I see" was Richard's response.

The Kente Cloth pCSDT offered the opportunity to explore the mental model of the expert crafts person, in order to gain an understanding of their perception of their craft. In the case of Richard, his mental model was so complex that we had to scale back our efforts to duplicate his exact understanding of the craft in our representation of it a software model. As mentioned above, this negotiation of fidelity to the craft representation, and usability by the student is not always easy to balance and sometimes takes enormous effort to 'get it right'.

3.5 Conclusion

As we have seen in this chapter, ethnocomputing requires that we focus on the various aspects of 'computational thinking' within the construction of cultural practices and indigenous craftwork. We have examined the history and development of the programmable Culturally Situated Design Tools as an example of ethnocomputing software. In particular, I have focused on the pCSDTs that I have worked on for my research, namely the Cornrow Curves, Kente Cloth, and Adinkra stamping. These Java applets have offered the opportunity to examine the development process of a set of robust applications from concepts to deployed applications specifically designed to teach 'computational thinking'.

We explored the process of eliciting the understanding of the craft we want to simulate in our development work using mental models. The craftsperson's understanding is likely to be very complex, especially if the craft itself is complex and practiced largely from memory alone. Gaining the understanding of a craft from the practitioner's point of view is extremely helpful when it comes time to create a representation in simulation. However, sometimes this mental model may be too complex for use in a simulation for students, requiring negotiation to find what we like to call the 'sweet spot' for our software. This is a place where the software that we build is faithful to the craft as practiced, where the simulation is tuned to an appropriate challenge for students, and where the simulation is of pedagogic value to instructors.

We have chosen to restrict many of the pCSDTs, at least in the user's initial encounter, to images that are appropriate in our effort of fidelity to the craft in simulation. Not all programming environments similar to the pCSDTs believe this to be desirable, in fact many do not. We call this 'anything goes' approach the content agnostic position and it is the subject of the next chapter.

4. CONTENT AGNOSTIC POSITION

4.1 Introduction

Since the introduction of LOGO in the 1960s, educators have been creating open-ended, "sandbox" style computational media; what Mizuko Ito has labeled the "construction genre" of educational technology [65]. Examples include MIT's *Scratch*, CMU's *Alice*, UC-Boulder's *AgentSheets*, and a wide variety of similar systems. Advocates of this approach, in particular Seymour Papert, frame these technologies as a means for allowing learners to teach themselves [66]. We are generally in agreement with Papert and his colleagues: our empirical studies, detailed below, suggest strong advantages offered by this bottom-up, constructionist framework, in contrast with the usual top-down, "instructionist" approach of many "edutainment" products. Our analysis also supports their contention that constructionism resonates with broader themes of self-empowerment and democratic politics [67]. However, we find that this emphasis on self-directed learning is often accompanied by an unacknowledged assumption that any technology which supports constructionism must effectively act as a blank slate or empty container; and that its designers must remain "agnostic" as to the content that learners create.

This "content agnostic" position, we contend, can have some disadvantages that are detrimental to students' development of computational and cultural literacies. Through a review of several case studies, we illustrate the possibility of alternatives that combine a "content aware" medium with a similar bottom-up, self-directed learning approach. Our goal is not to eliminate the content agnostic position, but rather to provide an opportunity for educators, designers, and theorists to reflect on its otherwise invisible presence within the construction genre, and to consider alternative approaches when appropriate.

Scratch serves as an exemplar for educational technology that is constructionist in design and deliberately content agnostic. Teachers can use Scratch to make class assignments that are not content agnostic, and ordinary users in the online Scratch

---

This chapter previously appeared as: M. Lachney, W. Babbitt, and R. Eglash, "Alternatives to the content agnostic position in the 'Construction Genre' of learning technology," *Computational Culture: A J. of Software Stud.,* submitted for publication.

community can group themselves together in "studios" that have some specific content in common. But the designers of Scratch have attempted to make the software itself as general and flexible as possible, and to impose as little influence as possible in terms of content. We recognize that this content agnostic position emerges out of positive goals: to nurture a sense of being a producer rather than a consumer; to encourage more genuine, intrinsic motivation for learning rather than extrinsic rewards such as game points; to give youth a voice from the bottom up rather than instruction from the top-down; and to help foster the general sense of free expression that is fundamental to an open society. Yet, we find that this approach tends to turn a blind eye towards the pervasive influence of social forces that can limit children's development and silence alternative material and perspectives.

We begin with an examination of potential disadvantages to the content agnostic position. Using case studies of individual Scratch projects, we will review four categories of potential disadvantages. In particular evidence for the strong influence of commercial products calls into question the stated Scratch claim of making students "producers rather than consumers." As a way of mapping possible alternative frameworks, we suggest an analytic separation between the two dimensions of interest: the spectrum between content agnostic and content awareness is on one axis; the spectrum between top-down instructionism and bottom-up constructionism is on the other. Since these can be treated as orthogonal axes, the impression that one can only choose between a content-agnostic approach and an instructionist system is a false dichotomy.

We then review possible alternatives, with a final focus on our experiences using Culturally Situated Design Tools (CSDTs: www.csdt.rpi.edu). While CSDTs also offer a bottom-up, constructionist approach to math and computing education, they highlight the cultural capital of indigenous and vernacular knowledge (Native American beadwork, urban graffiti, etc.) and foster their exploration as a computational medium. In contrast the content agnostic approach, in which such cultural references are typically restricted to the presence of a "content provider," the CSDT's "ethnocomputing" approach locates the computational thinking in the cultural practices themselves (for example the algorithms used by traditional native bead workers). Under what circumstances that may or may not be more effective in teaching computing and raising interest in computer careers is a

more complex set of questions; our purpose here is merely to "decenter" the content agnostic approach, which is often presented as the only possibility for constructionist learning, and show that there are viable alternatives that offer advantages which are rarely considered in the literature. We conclude that it is possible to maintain the tenets of authorship in constructionism--the valuable sense of free agency in knowledge construction and artifact building--while also supporting cultural connections that are often not readily available to students, and may hold particular importance for underrepresented ethnic groups.

4.2 Potential Disadvantage to the Content Agnostic Position: Four Categories

Potential disadvantages to the content agnostic position fall into 4 categories. We will list these from what we see as the category of least concern to that of the greatest concern.

4.2.1 Use of Inappropriate Material

We regard this as essentially a solved problem, insofar as any teacher allowing their students the freedom to write an essay, paint a picture or author a website would be faced with similar instances in which they have to weigh freedom of expression against offense to other viewers. Even the strongest advocate for a content-agnostic approach recognizes that some adult supervision and intervention is occasionally needed. The designers of Scratch have been admirable pioneers in developing scalable systems that allow peers, mentors, and/or teachers to "flag" inappropriate material, which often obviates the need for more intense forms of intervention and supervision.

4.2.2 Tendency to Gravitate Towards Violent Video Game Formats

We recognize that there are a wide variety of perspectives on this controversial topic. Perhaps the best empirically supported research has been by Olson et al. [68], an extensive study in the Harvard Medical School's psychiatric program, which concluded that there is little evidence for any increase in real, physical violence based on exposure to violent video games. We are less concerned with risk of violence, and more concerned

with the phenomenon of "gravitate." In the parlance of nonlinear dynamics, violent video games form a basin of attraction, which draws users towards their commercial material. There is a distinction between healthy play that may legitimately include violent elements as part of a "free-flow" creative repertoire [69], and the commodified forms of violence that commercial content encourages, which may take the place of creative acts that could make a better contribution to the users' personal and intellectual development.

Figure 1 shows an image from a Scratch user[1]. While the icon of a realistic gun image is no doubt inappropriate for a young user, this falls under category "a" above: use that will likely be eventually flagged as inappropriate by the community or by a Scratch system administrator. As shown to the right of that icon, the actual games this user has created are based on stick figures and other low-barrier animation components; they are so abstracted that it would be difficult to claim any strong resemblance to commercial video games. Despite the visual reference to a real gun and to "first person shooters" much of the activity resembles the ordinary "sociodramatic play" that is often cited as a psychologically healthy activity [70].

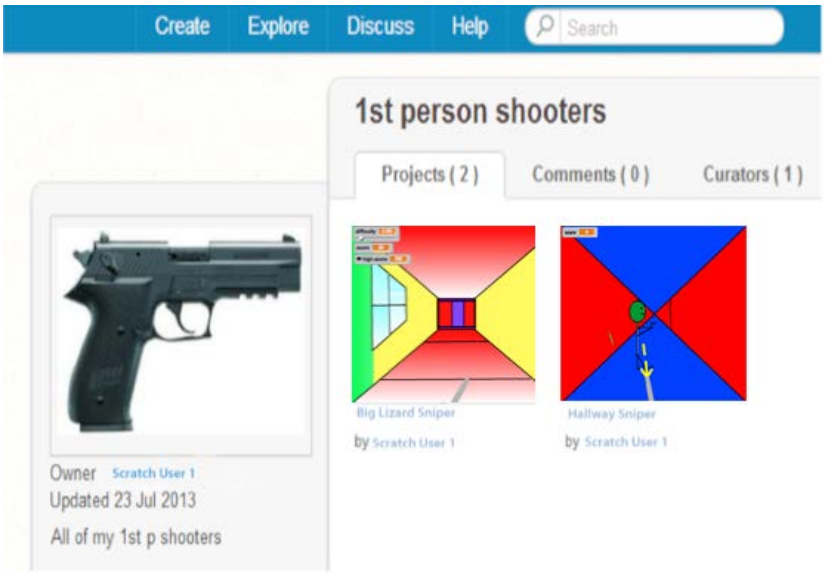


Figure 4.1: First person shooter projects in Scratch

At the other extreme, Figure 4.2 shows a series of games created by another user. These projects consist of graphic images imported from the commercial video game *Halo*. The difference in comments received by Scratch users with self-created images,

[1] We have changed the names and modified the images slightly to preserve anonymity

versus the utilization of commercial first-person shooter images from games such as *Halo* (particularly in the more realistic images such as the "Halo assault rifle" below), is striking. Responses such as "Nice work bro -- its beast" attest to the ways that both the *Halo* images and the language of response are recognizable tokens for a specific sense of masculinity that rewards these commodified forms of violent fantasy [71]. Although such networks of users may be only a small percentage of the total, we hypothesize that in such cases Scratch has essentially become "colonized" by the varieties of commercial video games that reinforce hegemonic masculinity, and thus may suppress healthier, alternative forms of masculinity (e.g. [72]). We now move to consideration of commercial "colonization" more generally.

Figure 4.2: Additional projects from Scratch users

4.2.3 Tendency to Gravitate Towards Commercial Content in General

The user in Figure 4.2 produces just a few of the hundreds of *Halo*-inspired Scratch projects; a brief search turns up dozens of users creating these works: Halo Zombie Defense 2, Halo reach firefight, Halo FPS shotgun, etc. One user alone lists thirty-seven Halo-related projects he created. These are not entirely negative: this appropriation and remixing of commercial content creates learning opportunities for computational skills and concepts, and the network of like-minded users offers motivation, camaraderie, and a social learning context. On the other hand, these

contributions also show generic conventions -- gender stereotypes, excessive militarism, etc. -- that stem from youth-oriented commercial culture. Beyond commercial gaming products, Scratch projects and studios yield a high amount of other media franchised brand name traffic: dolls such as Bratz and Strawberry Shortcake; popular characters from *Power Rangers*, *Care Bears* and *My Little Pony*, fast food outlets such as McDonald's Happy Meals, fashion items such as Adidas and Air Jordan sneakers, etc. A search for Barbie on the Scratch community site leads to a list of hundreds of animations, art, games, music, and stories all featuring the easily recognizable characters of Barbie and friends.

The reality is that these commercialized products, advertisements, and media commodities stands in strong contrast to Papert's idealistic vision: that constructionist learning media will allow ideas by children-designers to be born out of their own unique creative imagination, to come "from wherever fancy is bred [67]". Surely he did not have in mind that it would be bred in a corporate boardroom. In category 4.2.2 we noted that the problematic aspect of violent video games may not be violence itself, but rather its presence as a basin of attraction. Similarly, we recommend caution against a hasty conclusion that Scratch projects featuring highly commodified objects will always convey psychologically damaging or negative properties. Scratch projects do sometimes use irony or humor with these objects, and several scholars have documented the ways in which self-conscious semiotic play can subvert gender roles and passive conceptions of readers/viewers [73]-[75]. But the majority of the Scratch projects focused on toys, fast food, violent video games and other marketing schemes appear to be simple celebrations of the commodities, and even those that offer an ironic take are still caught in the basin of attraction that brings so many projects to focus on commodified cultural forms, at the cost of cultural alternatives that can better contribute to children's social and psychological growth.

Sociologist Beryl Langer refers to the products of global corporations such as Mattel, Hasbro, and Disney as "commoditoys" – toys that propagate their consumption meme across cartoons, movies, fast-food outlets, clothing, and a seemingly endless variety of add-ons, accessories, and other media [76]. Ferguson documents the ways in which Langer's commoditoys "implicate children in a collective trance, inspiring or

strengthening a subconscious belief in the mythic powers of capitalism [76]". One need not be against the concept of free enterprise to find something objectionable in the commodification of childhood. While constructionist literature correctly describes Scratch's advantage in "turning kids from media consumers into media producers," that characterization ignores a secondary level of consumption: they are now "producers" of what are often essentially commercials for commodities. It is this secondary level of consumption that is made invisible by the content agnostic framework.

4.2.4 Differential of Computational Complexity between Commercially and Non-Commercially Engaged Projects

In our review of Scratch projects, we have found that there seems to be an inverse relationship in computational complexity between projects that engage with commercial content and those that engage with alternatives, such as heritage or social critique. For example, the project "Black History," offers an animated overview of the Brown vs. Board of Education case. When we examine the scripting necessary to create the simple slide show used to tell the story, we find one script with a very small number of codelets, only one of which involves control flow ("wait" for timing the images). In comparison, the stick figure games in Figure 4.1 use up to 15 separate scripts with a vast intricacy of control loops, conditionals and other features. This seems to be a fairly consistent pattern: as if technical proficiency and culturally significant content are in a zero-sum game. But of course they are not so in the abstract, so why do they appear so in practice?

All four of the categories above--inappropriate material, violent video games, commodified content, and the reduction in complexity--are admittedly the result of the users' environment and complex social influences, not an inherent property of the software itself. Given this fact, we imagine that an advocate for the content agnostic position might reply that the goal of Scratch is simply to teach computing, and that it provides an unbiased platform in which users must bring their own personal goals and cultural judgments to bear on their development of computational literacies. Such arguments are essentially making an analogy to the concept of free speech in a political system: it is true that no thoughtful advocate for democracy would start placing

restrictions on free speech with the misguided goal of improving its content. But in our view, a better political analogy would be the issue of campaign finance reform.

Since its origins in 1867, many US legislators have proposed that we limit the amount of funding that the wealthy can use to influence elections. Such legislation is controversial; there are always complaints that restrictions on campaign donations will violate free speech. However, the Supreme Court recognized in Austin v Michigan Chamber of Commerce (494 U.S. 652, 1990) that the government did have an interest in protecting our democracy from the "corrosive and distorting effects of immense aggregations of wealth that are accumulated with the help of the corporation and that have little or no correlation to the public's support for the corporation's political ideas [77]." The strong influence of marketing and other dominant forces on Scratch is analogous to the influence of wealth on an election. The problem is not that free speech should be restricted, but rather the question of how "freely" ordinary citizens can be speaking, if the other side is empowered by the enormous resources of corporations: wealth, focus groups, advertising campaigns, market penetration, branding, and so on.

Thus, we need alternatives to the content agnostic position, one that can "level the playing field". As software developers, we cannot control exterior social influences, but we can modify attributes of the learning medium, as discussed below.

4.3 Constructionism and Contextualism as Orthogonal Dimensions

Our thesis is that constructionism and the content agnostic position have been erroneously presented as a single phenomenon, resulting in only two possible endpoints along a single spectrum. Once we separate these into orthogonal dimensions, new options arise. Figure 4.3 shows these separated dimensions using examples in math and computing education.

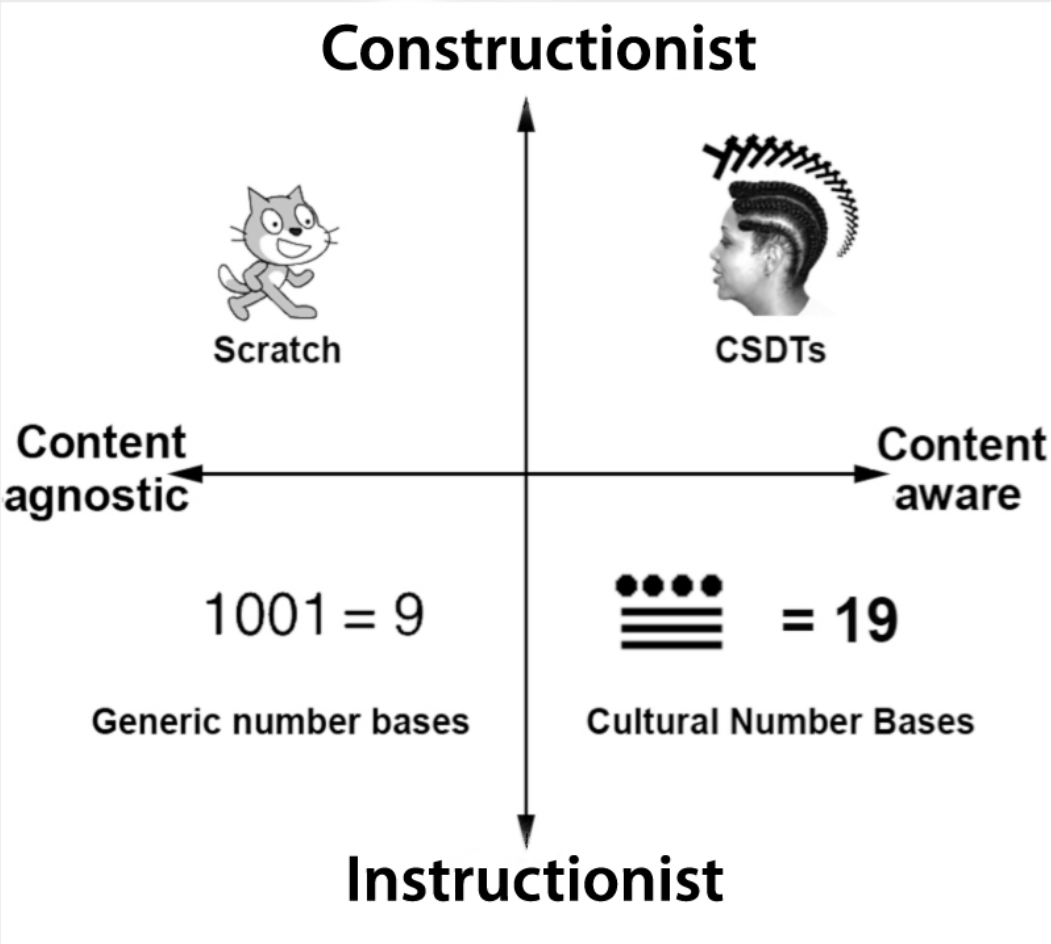


Figure 4.3: Two orthogonal dimensions

At opposite ends of the vertical axis we have the constructionist vs. instructionist distinction that has been the focus of Papert and his colleagues. On the horizontal axis, we introduce the spectrum from content agnostic to content aware. Thus, there are four possible categories, not just two. How did the impression that there are only two categories emerge from a broader history of the construction genre of educational technology?

From the development of the LOGO programming language in the 1960s and its evolution into "Turtle Geometry" [78], [79], to popular 21st century constructionist design tools [17], [80], [81], information technologies have been utilized as avenues to offer a digital sandbox for young children to develop mathematical and computational thinking. Much of the conceptual basis for this approach originates in the learning psychology research of Jean Piaget, who framed the development of cognition in terms of four main stages that build towards increasing levels of abstraction. Piaget's theory of

cognitive development describes learning in terms of assimilation of new information into current cognitive structures, followed by the eventual development of those structures as they "accommodate" from the repeated assimilations [82]. For Piaget, children must move linearly through four "developmental stages: sensorimotor, preoperational, concrete operational and formal operational [82]." However Papert and others have taken less dogmatic approaches to these stages (which is fortunate since they did not hold up in further investigations; e.g. [83]). We will follow the popular convention of educational technologists in referring to Piaget's original theory as "constructivism" and Papert's more pluralistic conception of child development as "constructionism."

According to Piaget, by the age of six children are able to think concretely about objects and things in their world, and to problem solve through processes of trial and error, but it is not until the age of twelve that the abstractions of formal thinking are possible [84]. Concerned with issues of "epistemological pluralism" Turkle and Papert [85] subvert Piaget's operational hierarchy of development. They posit a more flexible understanding of "concrete"[2] versus formal thinking as not fixed developmental stages but rather alternative and institutionally legitimate ways of knowing. Papert and Turkle draw on a variety of sources for this alternative. One is Evelyn Fox Keller's [86] study of Barbara McClintock's Nobel prize-winning genetics research, showing how the development of "a feeling for the organism" can sometimes reveal aspects hidden by abstract formalisms. Another is the feminist critique of moral psychology by Carol Gilligan [87], showing that girls' tendency towards contextual interpretation of rules is not a developmental lack but rather "a different voice." Most importantly, they described the work of K-12 and college students in programming and IT, showing that preferences for less formal programming styles were not necessarily barriers to technical success. In this way, Papert alters Piaget's biologically fixed developmental model in favor of a more bottom up model of scaffolded learning based on the access and availability of computational materials.

[2] Their terminology of concrete/abstract is perhaps misleading: in the case of different programming styles, both are pure code and in that sense equally "digital abstractions." A better characterization might be bottom-up -- structuring code organically by iterative experiments -- versus top-down; planning out a hierarchical decomposition of components.

Simultaneous with this emphasis on providing cognitive scaffolding came an emphasis on discovery learning. From the point of view of Papert and others, these are not two separate domains, because the essential meaning of "discovery" implies starting from a prior set of skills and knowledge, which would likely be more concrete, embodied or experiential. However, it is not the case, from Papert's point of view, that one can simply "mix and stir" concrete learning and bottom up discovery, as he describes in this passage:

> What can be done to involve the mathematically alienated child? It is absurd to think this can be done by using the geometry to survey the school grounds instead of doing it on paper. Most children will enjoy running about in the bright sun. But most alienated children will remain alienated. One reason I want to emphasize here is that surveying the school grounds is not a good research project on which one can work for a long enough time to accumulate results and become involved in their development. There is a simple trick, which the child sees or does not see. If he sees it he succeeds in measuring the grounds and goes back to class the next day to work on something quite different [88].

In other words, measuring the school grounds may be more concrete, but it lacks "recursive depth" [89]; the possibilities for self-generative expansion. Papert contrasts this simple measurement of school grounds with learning from pattern generation on the computer screen via LOGO, which "if worked on with a good dose of imagination, indicate the sense in which there are endless possibilities of creating even more, but gradually more, complex and occasionally spectacularly beautiful effects [78]." Thus, the computational medium becomes, in a recursive fashion, its own learning context. He makes this explicit in his 1980's introduction to the text *Mindstorms*:

> But to say that intellectual structures are built by the learner does not mean they are built from nothing: to the contrary, children appropriate to their own use materials they find about them, most saliently the models and metaphors suggested by the surrounding culture... When I speak here of "our" culture... I am

> not trying to contrast New York with Chad. I am interested in the difference between pre-computer cultures (whether in American cities or African tribes), and the "computer cultures" that may develop everywhere in the next few decades [78].

Thus, Papert's computational constructionism did not ignore cultural influence, but it did imply that African villages, the American inner city, or similar environments where computers were unlikely to be found in the 1970s were lacking potential for children's self-development of computational thinking. In contrast, any place with computers (no matter where geographically) could best support children exploring the endless possibilities by which simple patterns can elaborate upon themselves. The computer, in this view, is not merely a medium by which math and computing become exploratory tools, but also a purification device [90] by which we can "siphon off" the irrelevant aspects of African or American culture and allow children to directly experience the raw stuff of science; a direct yet playful engagement with nature's laws of math and computation. Indeed, a common critique of constructivist pedagogical models that maps on to Papert's approach to computer education is made by popular educators in the global south who argue its theoretical and practical oversights in the preservation of indigenous knowledge systems are based on Western assumptions about linear and industrial progress [91].

Papert's tragic accident in 2006 prevented his further direct involvement in constructionist technology development. As an editor for *Socialist Review* in the 1950s, he was not naive about the power of capitalism to colonize our lives, but his political commentary in "Perestroika and Epistemological Politics" [67] indicated that by the 1990s he saw the authoritarian tendencies of centralized government control to be at least as much a problem, and (rightfully) took pride in the resonance between bottom-up constructionist learning media and anti-authoritarian political critique. This critique is a powerful means to champion humanitarian principles such as the free speech guarantees of the first amendment, but difficult to apply to problems such as the imbalance between the speech of individuals and that of media corporations.

The *Scratch* learning environment developed not only from the influence of Papert's constructionist learning framework, but also the era of social networks and peer-to-peer media sharing. Michel Foucault uses the term "rarefaction" [92] to describe the phenomena by which the theoretically infinite utterances of an articulating subject are reduced to a remarkably predictable pattern that conform to the needs of dominant power structures. This process of rarefaction is another way to conceptualize the basin of attraction towards commodified forms of youth culture that is in contradiction to Papert's vision for an intellectually free and creative computational medium.

Thus the rise of a content-agnostic position, which envisions only a single dimension of instructionist versus content-agnostic constructionist environments, can be attributed to at least 3 influences. One is a vision of the optimum in computationally creative medium as that which supplies its own "cultural" content -- in the words of Sharon Traweek, a "culture of no-culture"-- in which nature's universal laws of math and computing can be playfully elaborated and scaffolded upon itself [93]. Another is the political atmosphere in which violations of human rights by centralized governments brought anti-authoritarian critiques to the fore. Finally, there is the fact that Scratch exists in a media ecology that offers its users easy access to the signifiers of commodified youth culture, an access not foreseen in Papert's original vision.

4.4 Content-Aware Learning: Examples from Culture-Based and Social Justice Based Math Education

In the case of Scratch, learning can take place without bias towards specific content because the scripting platform it provides is teaching computational thinking regardless of which images are used, which narratives are explicitly or implicitly supplied, etc. Such content-agnostic media is possible for many educational disciplines. Software such as Geometer's Sketchpad and GeoGebra can create any geometric form; Algodoo is similarly a general-purpose physics "sandbox." In contrast to these content-agnostic forms, "content-aware" learning media is designed to provide students with a kind of "value added" orientation or valence. Normative values such as social justice, environmental values regarding ecological sustainability, conscious consideration of the ways gender is encoded, and cultural connections, particularly to the backgrounds of

underrepresented students, are all ways in which designers of learning media can take a content-aware approach.

The idea of content-aware STEM education has become increasingly important in state and national education standards. For example, the Next Generation Science Standards, created through a partnership from the AAAS, NRC, and the National Science Teachers Association has recommended that rather than teach STEM education strictly though abstract universal principles, science education should include contextual connections, with "the goal that all students should learn about the relationships among science, technology, and society [94]". In particular they highlight the approach of González et al. [95], who demonstrated improvements in science education practices by incorporating local knowledge from home and community. The integration of content on sustainability with STEM education has also been a growing movement, as can be seen for example in the National Association of Biology Teachers' position statement [96]. The National Council of Teachers of Mathematics standards has emphasized that the "opportunity to experience mathematics in context is important" and thus "students should connect mathematical concepts to their daily lives, as well as to situations from science, the social sciences, medicine, and commerce [97]." The Computer Science Teachers' Association has similarly supported content aware approaches to computing [98].

The use of cultural, social and environmental context in STEM education is thus an important category for content-aware learning media[3]. As implied by Figure 4.3, it is possible to take both instructionist and constructionist approaches to content-aware media. A useful illustration of this distinction can be found in ethnomathematics. Although initially defined in the context of indigenous societies [99] it quickly expanded to include ancient non-western state societies (Egyptian hieroglyphics for example), mathematics in vernacular culture (e.g. calculations by push-cart vendors; quilters, etc.), and even historical and sociological investigations of professional mathematicians.

One variety of instructionist versions of ethnomathematics pedagogy can be classified as "number base systems." Different cultures use different bases for counting; for example Mayan hieroglyphics show a base-20 system with a sub-base of 4, and this is

[3] It is not the only category possible.

easily converted directly into lesson plans [100]. That is not to say that it is impossible to develop a constructionist approach using this material, but a quick Internet search will show an enormous number of "Mayan math" worksheets available to teachers in which students are simply translating between our decimal (base-10) system and the Mayan vigesimal (base-20) system. A similar category is that of culture-based word problems. As in the case of using number systems from different cultures, using word problems from different cultural scenarios also runs the risk of trivialization; replacing Mary and Susie counting marbles with Juan and José counting bananas [101]. Zolkower describes her frustrating experience in attempting to interest New York students of Puerto Rican heritage in math word problems that referenced ancient civilizations of South America, under the presumption that both qualified as "Latino" [102]. The presence of these instructionist approaches to culturally specific content is surprisingly common; the ubiquity may be partially due to the fact that it requires very little modification of standard math lessons, and little distraction or additional labor for the already-overburdened math teacher to add on this kind of shallow "cultural" veneer.

This shallow gesture towards "multicultural math" bears little resemblance to scholarly ethnomathematics research. For example, Ascher's [99] description of the Native American game of Dish goes beyond a mere cultural example of probability. In the Cayuga version of the game six peach stones [103], the stones are blackened on one side, are tossed in the air. The outcomes are recorded by the number of times the peach stones land black side or brown side up. The Cayuga traditional point scores for each result rounded to the nearest integer, are inversely proportionate to their probability: it appears as if they were calculated. Ascher does not claim that is the case; rather she notes that this accuracy can emerge, as Eglash relates, because of how the game is embedded in community ceremonies, the people's spiritual beliefs, and their healing rituals; especially through their concept of "communal playing". Rather than attribute winnings to individual players, this communal playing attributes them to the group [103]. Another ethnomathematics analysis for this accuracy in Native American probability [104] is the figure of the "trickster" who acts randomly in myths and legends, and relates randomness in games of chance to horticultural practices that maintain a diversity in plant genetics to match the randomness of environmental fluctuations. Thus,

ethnomathematics research offers a correspondence with western knowledge of probability while situating it in relation to a rich set of cultural practices, rather than reducing it to a decontextualized algebra or word problem.

However, translating this research into classroom lessons is a difficult challenge. If a student is already alienated from classroom work, expecting them to read a complex text to gain an understanding of these deeper connections is unlikely to succeed. Thus, one advantage of constructionist approaches to ethnomathematics is the possibility of using the creative, engaging elements that Papert and his colleagues have rightfully emphasized to allow for this deeper engagement with underlying cultural and technical meanings. Lipka et al. for example developed a set of culture-based lessons for native Alaskan students, which fully embraced both constructionist and content-aware approaches. In the control group, observing standard lessons as they had always been taught, they found that "the students were to find the perimeter of various geometric shapes (e.g., a girl, a butterfly, a house) on graph paper... mostly in silence [32]." There was also an opportunity for the students to calculate the perimeter of their own, hand-drawn figures.

In contrast, in their experimental group they observed that in the exercise on measuring perimeter in the culture-based *Building Fish Racks* module "the children were able to see that only certain sizes of rectangles would be appropriate for a fish rack. They knew this because they saw fish racks in their village on a regular basis [32]". Their constructionist approach to perimeter concepts made use of this cultural knowledge to scaffold meaningful measurement activities; it had "recursive depth" in the sense that prior experiences became the basis for successive activities and explorations in the geometry of design and measurement, which include a new appreciation for the role of mathematics for indigenous culture (for example finding out how elders used body parts as measurement units). As a result there was statistically significant improvement in pre/post test score for this experimental group in comparison to their control group [105].

It is worth comparing this to the earlier quote from Papert regarding a similar example of measuring the dimensions of artifacts in local surroundings, which he criticized: "surveying the school grounds is not a good research project on which one can work for a long enough time to accumulate results and become involved in their

development [78]." The phrase "involved in" gets to the heart of the matter. If measuring the school grounds is insufficient for this deeper sense of involvement, then what is? There is no doubt that commodified content can create a sense of deep involvement: video game enthusiasts show their life-long brand loyalty with tattoos for *Grand Theft Auto,* and "shopping haul" videos in which consumers showcase their purchases has been one of the fastest growing trends on YouTube [106]. However, this is not the kind of "involvement" that contributes to the social, ethical, and intellectual strengths of a developing child. An awareness of the content is needed, even if that is simply guidance from a caring instructor. But it is also possible to build that content awareness directly into the medium itself.

A content aware approach to computer literacies can aid young people's cognitive and emotional development of community and self; including a healthy understanding of ethnicity and racial identity [107]. Healthy self-conceptions can vary greatly; from students who see ethnicity as central to those who do not. There is however, substantial evidence in developmental research and experimentation to suggest that negative racial and ethnic stereotypes have significant impact on student performance. Altschul et al. [108] demonstrate that the connection one feels to their immediate racial-ethnic community, awareness of the racial attitudes of others, and the larger ethnic in-group community's attitudes and values towards school are significant indicators of academic success. This is to say, when negative stereotypes of a group exists and are not actively resisted, salient group membership is "disruptive to performance" [108] -- but there are a wide variety of resistance strategies. A dynamic view of racial and ethnic identity, and media that can accommodate the wide variety of strategies and heterogeneous conceptions of race [109] is thus better taken to avoid culturally deterministic educational practices.

Another reason for using a culture-based approach is the myth of genetic determinism; the false conception that underrepresented students are incapable of academic performance at comparable levels as their peers because of genetic differences in neural structure [110]. There is no evidence supporting this, and much to the contrary. For example, after the occupation of Germany at the end of WWII there was a population of illegitimate children of black and white soldiers who were raised by their

white German mothers [111], found no black/white IQ differences in this group: not surprising since they were raised in identical (German) environments. More recently similar results were found for black children raised by white mothers in the US [112]. But the myth is not harmless: studies of "stereotype threat" do show that African American students fair worse on standardized tests than their white counterparts, when they believe the test might be reflecting some ethnically determined intelligence [110]. In other words, the myth of genetic determinism can result in a potent self-fulfilling prophecy. Demonstrating complex mathematical and computational thinking as part of the heritage culture for underrepresented students is a potential counter to this pernicious falsehood.

That is not to say that connections to heritage culture are the only legitimate path to this sense of deep involvement. Vernacular culture such as graffiti, breakdancing and similar practices can also be part of youth-produced creations. They too are subject to commodified forms — in today's world what isn't? — but it is not difficult to train youth to recognize the distinction. Providing students with the tools to analyze forms of commodification and exploitation can be just as important to the constructionist repertoire. For example, Eric Gutstein's work over the course of two years in Latino middle schools in Chicago applied a Freirean approach to problem-posing pedagogy in mathematics [38]. The content of Gutstein's work emerges out of students' lived experiences of gentrification and racial/class segregation in their neighborhood using the constructionist goals of open-ended learning. As Freire explains, "problem-posing education affirms men and women as beings in the process *of becoming*—as unfinished, uncompleted beings in and with a likewise unfinished reality [113]".

As Gutstein's classroom and others become spaces to perform and learn about social justice through mathematical modeling, students begin to form their own questions and conclusions about what parts of society are equal/unequal, racist/not racist, just/unjust, etc. A hallmark of constructionist learning is that the students are free to contradict the teacher, and indeed Gutstein notes that students often came to conclusions that contradicted his views: "educators need to be explicit in their views while at the same time... [respecting] the space for others to develop their own [114]". At the same time, Gutstein acknowledges the delicate balancing act between offering students research topics that bring issues of social justice to the fore, and the potential for discouraging

them if they feel the problems are too overwhelming [115]. The goal is a learning process which "poses to students their life condition not as immutable but merely as challenges... upon which people can act and change [38]".

4.5 Content-Aware Constructionist Learning in Computer Science Education

As in the case of Gutstein's work in mathematics, it is possible to use a constructionist approach to social justice-based computer science. Ryoo et al. [116] and Scott and White [117] describe curricula in which underrepresented high school students used a variety of electronic media to investigate -- in open-ended, constructionist fashion -- issues at the intersection of social structures (the "subject positions" of race, class and gender) and quality of living for groups organized by those structures (health, employment opportunities, etc.). For example, Ryoo et al [116] noted that in one students' reflection on her health game she "described her realization that her aunt's obesity was affected by complex factors beyond diet and exercise, including the intersection of her aunt's low-pay/low-status job, high crime neighborhood, inaccessibility to healthy food options, and her family's material needs as contributing factors in her struggle toward a healthy lifestyle [116]" (quoting from Lee). As in the case of Gutstein's social justice-based math education, "the program impact was because of the culturally responsive practices (asset building, reflection, and connectedness) embedded within the curriculum [117]", not because of the digital media itself. This is by no means a flaw -- surely committed teaching by well-trained instructors is the ideal -- but it does create challenges for scaling up such programs.

It is possible, however, to build a content-aware approach directly into the underlying structure of constructionist learning media: such media would have the potential advantages of easier scaling (like any web-accessible media), facilitating the ability of instructors to add the desired social values, and -- even in the case of users without instructors -- it could offer a counter to the forces of commodification critiqued earlier. This is the case for the suite of applets we have designed and tested, Culturally Situated Design Tools (CSDTs). Similar to the concept of ethnomathematics, the basis for this "ethnocomputing" [118] is the idea that there is already computational knowledge

in these cultural practices: iterative patterns in Native American beadwork; recursive[4] applications of transformational geometry in African American cornrow braiding patterns; polar coordinates in urban graffiti, etc. In the case of indigenous knowledge systems there is potential for opposing the myth of genetic determinism and, in both cases, the potential for opposing the myth of cultural determinism.

The process of creating CSDTs begins with interviews with artisans, videos of their practice and "reverse engineering" of their designs; these are used to create a quantitative model that attempts, as closely as possible, to reflect the cognitive and behavioral processes of the artisans in their social context. This is important because simply imposing computational thinking externally would not have any impact on the myths of genetic determinism (one cannot argue for the presence of computational thinking prior to colonialism) and might also detract from its effects on myths of cultural determinism. In the case of Native American beadwork, for example, we found that the concept of two orthogonal axes embedded in the rows and columns of the bead loom resonated with deeper cultural themes that were also organized by four-fold symmetry: native languages using base four counting; teepees made with four base poles; prayers offered to "the four winds;" etc. Interviews with artisans also revealed the use of iterative patterns; "up one over one" to create a 45 degree angle for example. The resulting model -- iterative patterns on a Cartesian grid -- is not something a beadwork artisan would immediately tell you (in fact, most artisans begin these discussions by either saying "it can't be explained, you just have to learn it with your hands"). But neither is it merely imposing alien math and computing. It is, rather, a sort of "composite picture" of the web of computational thinking that is in both the cultural background and the individual artisan's thinking and behavior.

The next step is in creating a graphical interface in which these indigenous or vernacular concepts and practices can be easily manipulated to create the traditional patterns. Some compromise has to be made between faithfully reproducing the artisan's concepts and behaviors, creating an interface that is easy and intuitive for children, and satisfying the teacher's need for relevance to the curriculum. In the cornrow braiding

[4] Here we are using "recursive" in the sense of "circular feedback of information" rather than in the coding sense of calling a procedure that calls itself. Whether or not such circular information flow is implemented as recursion or iteration in the coding structure is beside the point.

simulation, for example, we need to use the parameter of "translation"--a term from the standard school curriculum -- to set the distance between each of the Y-shaped "plaits" that make up one braid. But since the plaits are usually scaling in size as you move along the braid, the translation distance has to scale as well. Therefore, this CSDT sets translation as "percentage of image width" -- that way students do not have to create a separate variable and modify its magnitude, which would be a significant barrier (a "steeper learning curve") to new users. Interestingly, braiding stylists also place the distance between plaits using a visual estimate of plait size ratios, so quantifying this as percentage actually brings it closer to the artisan's "emic" view. It was striking seeing this connection in action when we observed that students with prior experience doing real braiding sometimes began their simulation using a small plait and scaling by a percentage greater than 100: students without that experience always scaled down. Since many students did not know that it is possible to have a percentage greater than 100; this was a helpful illustration for how cultural knowledge could be leveraged as mathematical understanding.

In much of the literature on multicultural education (including ethnomathematics), the motivation is described by what we might call the reflection theory: the need for creating lessons in which the student's cultural identity is reflected in the math or other content. Millerick for example suggests that, "by framing instruction to align with students' cultures, teachers can use curriculum that honors each students' life experiences [119]". While that is true in some cases--the above example of prior braiding knowledge for example--we have not found the reflection theory to be a good framework for understanding the variety of learning possibilities and potentials in our experiences with CSDTs. Evaluations have shown statistically significant increases in controlled studies of both math and computing skills as well as interest in computing careers [120]. But given a choice between the CSDTs from a variety of cultural origins -- fractals in African architecture, breakdance movements, etc. -- students do not show a strong correlation between their heritage culture and the tool they select. How to explain this contradiction?

One problem with the reflection theory is that it frames each student's identity as singular, unified and static. This is a poor model, as identity is constantly in a process of being constructed, especially in youth. Anthropologist Bourdieu provides a better

framework with the concept of "habitus," which allows for multiple proclivities, habits, expectations, etc. that constitute identities as dynamic and multidimensional. An individual's habitus, in Bourdieu's formulation, is still strongly influenced by social structures -- working class kids learn to value blue-collar ways of being -- and the social structures are in turn created by those individuals. In this aspect it is similar to Willis's [121] classic on "how working class kids get working class jobs." But Bourdieu also provides a theory of social change: "the systematic exploration of the unthought categories of thought that delimit the thinkable and predetermine the thought [122]". Thus habitus leaves open the possibilities for agency, interpretation and contestation: what is sometimes referred to as "reflexivity" [115]. This combination of habitus and reflexivity holds up well in empirical studies of youth identity [123], [124].

Stressing reflexive freedom too much makes cultural identity seem trivial or too easily malleable. Thus, the content-agnostic position implies that the identity of "consumer" is easily transformed to that of producer -- "Scratch turns children from media consumers into media producers" -- despite the evidence that much of the children's construction is filled with forms and practices of commodification. At the same time, stressing deterministic, unitary models of identity--as we have found in some of the literature on multicultural education -- also leads to poor predictions. Viewing identity as more multidimensional and flexible -- a constant negotiation between creative invention and structural influence -- helps to explain why underrepresented youth might respond positively to a broader array of cultural connections. A website that makes the case for sophisticated math and computing concepts from African, Native American, Latino, and urban vernacular practices is in itself a kind of symbolic representation of an anti-racist stance. As a set of design tools, it offers the attraction of what Pollock [125] terms "everyday anti-racism": not the heroic role model of Martin Luther King but rather a supportive environment in which mixtures of agency and identity can be creatively explored and developed.

Bringing together the "pluralist" constructionism of Papert with Bourdieu's concept of habitus helps to illuminate how a content-aware constructionist medium like CSDTs can facilitate underrepresented students' capabilities in bringing together computational elements, cultural hybrids and technological appropriations. For example,

Navajo Rug Weaver is a CSDT that draws on the "embedded" geometry and algorithmic processes found in the weaving and design of Navajo Rugs [21]. Like Scratch, students develop computational capital as they drop in and build with codelets to perform functions that aid in the design of Navajo rug patterns. Yet, unlike Scratch, students who use the CSDTs couple the acquisition of computational capital with explicit expressions of their own cultural capital. It is our opinion that making students aware of the cultural capital that they already possess but of which they may be unaware, is the important difference between the content agnostic and content aware approach.

Furthermore, Navajo Rug Weaver is designed to be appropriable by students beyond the imitation or even variation of traditional designs. Like the rest of the CSDTs, a software development and pedagogic goal is to emphasize its flexible design aspect in ways that "allow students to utilize a synthesis of math, computing, and culture in creative expression [120]". While the Navajo Rug Weaver obviously uses situated content, the software enables students to engage in self-making design activities that extend beyond the situated content and rugs themselves. For example, consider a group of high school students using the Navajo Rug Weaver. Some of the students' designs were imitations or variations of the weaves. Students' finished products reveal the importance of their own culture for motivating the design process. One students' description reads, "I wanted to get the 4 colors of the directions so once I got them I added the four hills. I added the purple background to represent my clan, water flows together" (Figure 4.4).

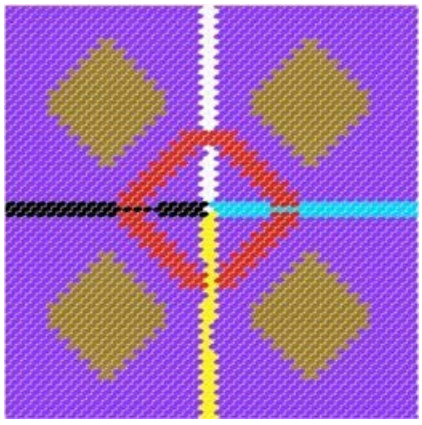


Figure 4.4: Simulation by Navajo student showing use of traditional motifs

The design and narrative indicate that this student is able to build with the CSDT platform in such a way that nurtures the cultural capital he shares with his community, while exploring its computational capital; thus fostering computational literacies. However, it is important that the tool allow students to go beyond the imitation of the situated content. Other student designs reveal the appropriation of the tool for interests that are explicitly not part of Navajo culture. Without this ability, the tool would be vulnerable to charges of essentialism, rendering CSDTs as culturally deterministic. Another Navajo student, for example, explained how she used the Rug Weaver to make a design using the colors of the Jamaican flag. Yet her design still made use of traditional symmetries.

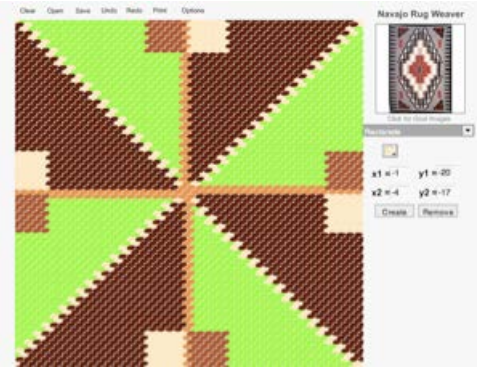


Figure 4.5: "Jamaican Summer Nights"

This example helps to illustrate the synthesis between the guidance of situated content and the openness of the construction genre. First, students are given a sandbox like platform that allows them to build artifacts and designs from the bottom up through what Eglash et al. [120] call “deep design” - the exploration and discovery of indigenous math and computing practices in social context. We previously discussed this in the case of four-fold symmetry in the virtual loom: not merely a trivial consequence of practical necessity, but rather a pervasive cultural theme that resonates with cosmology, healing, architecture, etc. Students review this cultural connection before using the applet; thus, the Navajo weave simulation is carried out in a context supporting these deep connections between math and culture. The fact that the student above chose to apply the colors of the Jamaican flag shows that it is sufficiently open-ended; the fact that it is the Jamaican flag and not, say, the colors of a McDonald’s sign, suggests that there is a kind

of alternative basin of attraction around culture, just as Scratch has a basin of attraction around commercial content.

Finally, we note the work of one Navajo student whose design description indicates a reflexive and critical turn. She describes the design in Figure 4.6 as follows:

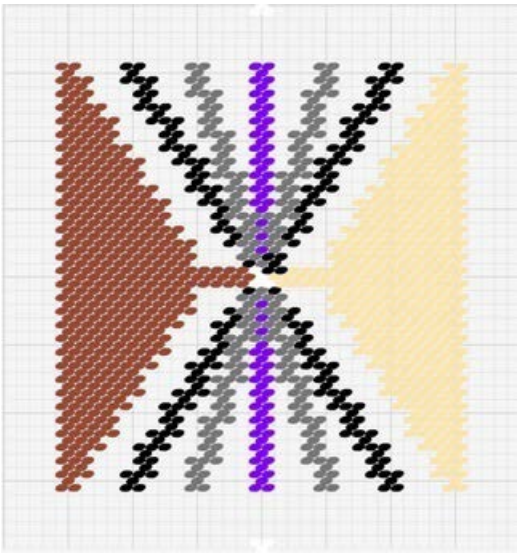


Figure 4.6: "Clash of Civilizations"

> The Clash of Two Worlds represents the western civilization coming in connection with the Diné Culture. When the two worlds collide in the center, it is an intermingling of cultures. Almost bliss, like Nirvana. A sharing. An understanding or an invasion? The connection creates an explosion that neither worlds can control. Our secrets burst away from us. Our culture, our respect, integrity, morals, our life explode in every direction. Leaving an emptiness [58].

On the one hand, a critic could regard this as an example of failure; a case in which the culturally situated approach is revealed to be discomforting or exploitative. But discussions with the students did not reveal such views; students were enthusiastic about the idea that Navajo traditions include mathematical and computational thinking. Thus the design is better understood as adhering to Papert's original vision: constructivist learning should be an opportunity for free speech, which by definition would include the ability to make critiques of the speech platform itself. It is perhaps analogous to an American artist who reminds us of the meaning of constitutional rights by burning the American flag.

Thus, the explorations are not limited to the situated content itself. Students continue to design in ways that meet their own creative ends and goals. Moreover, students are actively involved in creating negotiations between their own knowledge structure and that of indigenous math and computational knowledge. This negotiation enables a *dynamic* view of culture as something that is continually changing and being worked on at the everyday level. While CSDTs are content specific, they are by no means a static tool that drives students into creating homogeneous deliverables. They foster students' sense of "design agency" in open-ended negotiations between people, nature, and computers [54]. Both CSDTs and the work of Lipka, Gutstein and others offer avenues to rethink the regime of content agnosticism that currently subsumes the construction genre. As the CSDTs demonstrate, considerations of content aware educational technology does not necessarily indicate instructionism; there is a third alternative in which it can support explorations of historically and culturally relevant material in a constructionist way.

4.6 Conclusion

Without a doubt, the rise in the construction genre of children's technology and software enables teachers to expand and legitimize epistemological diversity in the classroom. Scratch and other platforms provide students with open-ended design tools that motivate authorial engagement with math and computational thinking. Yet, the coupling of constructionism with a content agnostic position can result in a basin of attraction for commercial content. The enormous amount of content involving violent video games, fast food outlets, commoditoys and other commercial products can be viewed as obstacles to Papert's original goals of bottom-up autonomous learning. In other words, corporatism has such a deep reach into all facets of youth culture that the content agnostic position allows the construction genre to be colonized by those entities. This should be actively confronted in critical and creative ways.

This is not to say that content agnosticism is always negative, only that the gravitational pull of commercial culture needs to be recognized as a significant force in the lives of children, and that we need not leave them unarmed in resistance to its tug. Educators, software and technology designers, and others should not give up the authorial

and democratic learning goals that surround the construction genre. In our view, these goals continue to provide important trajectories for re-thinking education in the 21st century. This is especially important in the age of No Child Left Behind and Race to the Top, which mandate a shift toward a national curriculum that results in homogeneous styles of teaching and learning, while demanding 'accountability' be measured by means of standardized test scores. In addition to technological scaffolding of democratic politics in the classroom, attention to content is a way to situate learning to create culturally rich and relevant lessons for students.

We can learn from critical pedagogies in ways that acknowledge the lived content of students' socio-economic and political realities, and draw on innovations such as ethnomathematics and ethnocomputing to offer exciting opportunities that draw on students' heritages and youth cultures in content specific ways. Balancing this content with open-ended constructionist goals, as with the CSDTs, invites students to creatively engage the connections between STEM knowledge and cultural knowledge, so as to make their own cultural capital more available to them as "things to think with"; and thus develop new ways of extending this capital. CSDTs are just one of many possible ways of approaching the future of constructionism in ways that enable an evolving view of inclusive educational practices. It is important that constructionism, in both theory and practice, continue to reinvent itself as much in its aspirations for positive social impact as in its technological sophistication.

5. FORMATIVE DATA ANALYSIS

5.1 Introduction

In chapter 3 I described some features of the initial iteration of pCSDT design process. In this chapter, I detail the user experiences with the programmable Culturally Situated Design Tools as they relate to our formative data analysis, and describe how that feeds back into our development process. It is important to note that observations of students and developers' experiences with the pCSDTs were collected under a protocol approved by the Rensselaer Institutional Review Board (#998), and with the consent of all parents/guardians, the consent of the schools that allowed the data collection, and the assent of all students involved in this analysis. In our formative evaluation, we primarily use a qualitative analysis of these developer and user experiences gathered by the use of ethnography.

Each of these tools -- Cornrow Curves, Kente cloth and Adinkra stamping -- presented unique challenges to the software developer working on the design and implementation of the program code. First, as we shall see in the ethnographies, meeting the fidelity expectations of the representation of the craft or practice was especially challenging. The development challenge involves negotiating what are often two opposing forces. First, we want to meet the expectation that the simulation will present the craft or practice as it is done in the physical world, and perhaps more importantly, as conceptualized by the artisans, since the purpose is to convey how this indigenous practice embodies indigenous math or computing concepts. Second, we need to keep the interface tuned to a level of usability that prevents students from becoming frustrated while using the software.

Another challenge encountered during the development effort of the Cornrow Curves, Kente, and Adinkra simulations involved design decisions about the extent to which the software should be flexible and extensible. It is vital that we make the correct

---

Portions of this chapter previously appeared as: W. Babbitt, "An analysis of the programmable Culturally Situated Design Tools from an HCI perspective," presented at the *3rd Annu. Symp., Theory and Research in HCI*, Troy, NY, USA, 2012.
Portions of this chapter previously appeared as: B. Babbitt, D. Lyles, and R. Eglash, "From ethnomathematics to ethnocomputing," in *Alternative Forms of Knowing (in) Mathematics*, S. Mukhopadhyay and W.-M. Roth, Eds., ed Rotterdam, The Netherlands: Sense Publishers, 2012, pp. 205-219.

decisions about how flexible, extensible, and to what degree of complexity the user interface provides for the student. From a birds eye perspective, we want the software to be easily approachable, that students can begin working on creating a simulation immediately, with a minimum of instruction, as Resnick describes with his 'low floor' metaphor [48]. At the same time, we want the software to be flexible enough that students to do not get bored quickly in using it, and extensible enough that it facilitates user creativity, Resnick's 'high ceilings' notion [48]. All the while, we want to try our best to avoid the pitfalls of the commercial colonization of the user space that we detailed in chapter 4.

Finally, we show how the program design gradually takes shape through the "Agile" iterative development model. The agile method that we examined in chapter 3 has been crucial to the success of the project. Each design tool is quickly made into a working prototype in the software development process; this enables data gathering and user feedback to be collected early in the work, to feed back into the design process. This approach reduces the amount of valuable time pursuing solutions that end up either not working or prove to be a bad fit pedagogically in the solution.

5.2 The Use of Ethnography

The use of ethnography in software development and human computer interaction (HCI) can mean different things to different people, and the term can vary in both scope and breadth in its use when we talk about HCI relationships. Using an ethnographic approach in HCI design can mean to gather qualitative information about the tasks that users perform with a computer system [126]. Using an ethnographic approach in HCI design can also be about the way in which researchers approach the gathering of this user information. Furthermore, it can be about how individual users perform specific tasks. In general, we can consider ethnographies to be ‘stories’ that recount the experiences of the user’s interaction with a computer system that can then be used to understand and improve the use of that system. For developers, ethnography represents 'stories' that relate how the software came to be as it is, detailing problems and approaches to solutions for those problems.

Gathering ethnographic stories for both the use and design of the pCSDTs has required keen observation. Verbal statements may not always match what the developer or the user is actually feeling or thinking [126]. The study participant may not be able to adequately communicate his or her feelings or thoughts to the ethnographer in a way that will be understood. In this ethnography work, we have attempted to adopt the participant-observer role [126] where we became an active participant while recording observations. The goal of these participant-observer ethnographies has always been to better inform design decisions to improve the stability, usability, and pedagogic value of the pCSDTs.

5.3 Ethnographic User Stories

The ethnographic stories gathered from the users of the simulation software can focus in different areas. Ethnographies that focus on the scripting interface will provide us with valuable information concerning the efficacy of the tool in teaching computer science topics such as iteration, conditional program flow and algorithmic thinking. Ethnographies that concentrate on the output window, will allow us to infer if the tool is effective in knowledge transfer by answering such questions as 'did the user produce an artifact that demonstrates the use and understanding of the mathematics or computer science concepts?' Ethnographies on specific portions of the software interface can tell us about these different software elements depending on where our interest resides.

Stories gathered from user experiences provided crucial information in determining the appropriate level of complexity for the scripting interface. For example, an early question that arose in the development process was how much of the calculation for generating a logarithmic or linear curve on the screen should be shown to the user, and how much should just simply be handled "behind the scenes." One possibility was to put a disk image on the screen and then leave it to the user to write a script that iteratively overlaps disks into paths of curves. However even using our own team members as the "users" (especially those coming from a social science background), we found this to be too challenging. Had we chosen that design trajectory, the script to create the curve would have needed to involve the use of variables and complex calculations. It quickly became clear the complexity of the 'dot' approach was not desirable from a user point of view, and we opted for a codelet that placed a spiral on the screen with 'in-place' values

that could be easily adjusted to create the desired image. This example demonstrates the types of decisions made during the development of the interface. These decisions were often difficult, but it was vital that we get them right, as the software would succeed or fail based on these choices.

From a development point of view, these user stories were instrumental in both helping to reduce user frustration and fixing garden-variety software 'bugs'. The narrow line of frustration versus challenge is not always easy to see, and frequently can only be seen through the eyes of a user. Users typically do not have the familiarity with software that a developer does, and this developer familiarity often leads to developer 'blindness'. Examples of software 'bugs' that user experiences reveal can be unexpected, from 'the spiral generates in the wrong direction', to 'the starting angle is off by 90 degrees'. These types of issues, of course, are all a matter of perspective and are likely to come to light through the observations of someone that has not been looking at the software for many hours.

5.3.1 Cornrow Curves Simulation with Two Students

This user story highlights Jackie and Tomas[5], two students who attend a 'high needs' middle school in Albany, New York. This middle school is located in an urban area, with about 650 students in grades 6-8. The student population is about 55% African American, 17% Hispanic, and 9% multiracial, with 72% eligible for free lunch. This school faces the daily challenge of educating students whose experiences at home range from stable to homeless, with parents missing from the household because they are in prison, drug rehabilitation, etc. Often, these challenges at home require these students to take care of younger siblings, such that homework and studying for school exams take a lower priority. The Cornrow Curves software was used by members of the seventh grade science club, which was composed of students that had chosen to participate in science enrichment activities.

These students worked with the Cornrow Curves for about an hour, during which time I recorded observations concerning their use of the program. I paid particular attention to how many objects (in the form of curves) they created and the complexity of

[5] The names Jackie and Tomas are both pseudonyms used for the purpose of this written account.

the pattern they were able to produce in that period of time. In addition, I recorded my perception of their reactions as they made scripts with the program building blocks, called codelets, to accomplish drawing tasks that interested them.

At the beginning of the work session, I provided them with a brief demo of how to simulate a braid. I first used a script, followed by an overview of how the software functioned. I then started with how objects were created (in this pCSDT, there is only one type of object, the plait, but the user can create multiple instantiations of that object). Next, we reviewed how the scripting panel worked, and the expectation that each script should start with an "On Begin" Event codelet. Finally, I explained that once an object is created, the codelet panel fills with all the available codelets for the object.

Thus, I instructed, "First place an 'On Begin' Event codelet in the scripting panel, then click on the Methods panel and add method codelets to define the plait pattern you want to create. For example, to create a new curve, click on 'Create New Object', choose 'Plait' and then begin selecting method codelets to complete the curve definition."

I then demonstrated the process by adding some codelets and then clicking "Begin", so that the students would know how to run their scripts and see the results that they produced. Finally, I demonstrated deleting an object from the Object Pane by right clicking the object to be deleted and choosing "Delete". After this brief summary on object creation, deletion, and script building, I encouraged the students to give it a try.

The students began working with the software, individually, on the netbook computers. For the remainder of the trial time, I did my absolute best to not interfere unless a student had forgotten to use an "On Begin" Event codelet at the top of their script, and only if they seemed unable to resolve an issue themselves. That is because my goal was to examine how well the students would do with a minimum of instruction: an important goal given that math teachers generally do not want to invest a great deal of time in training students on software. In wandering from student to student, I did occasionally ask, "What were you trying to do?" if something apparently unexpected had occurred, but otherwise I just simply praised them on the work that they were doing as general encouragement.

During the course of my observations, I noticed that student ability in working with the software ranged from having great difficulty with the programming process to

working with relative ease. I will focus on two students at opposite ends of this spectrum; their pseudonyms are Tomas (male Latino) and Jackie (female African American). Tomas demonstrated a fair amount of proficiency in working with the programming aspects of the software, and later mentioned some previous experience with programming. Jackie was at the other end of the spectrum.

Tomas quickly created the scripts necessary to generate a curve on the screen. He had little difficulty in navigating the interface to find the event, control, and method codelets that were necessary to accomplish the task and only once, when I happened to be near him did he ask a question concerning loop creation. Once I reviewed with him how to insert the variables in the control structure for a “do while” loop, he proceeded to complete his script and clicked “Begin”. I heard an audible gasp from Tomas, and upon returning to him I found that the result he was expecting was not what was displayed on the screen. I asked him “What were you trying to do?” He explained how he wanted the loop to function and how he wanted the plait to be drawn across the screen. Upon closer inspection, I realized that he had placed the “Duplicate” codelet before the Repeat-While loop, but I did not give him the solution. Rather, I suggested that he go back through his script step-by-step and see if he could figure out how to fix it.

Jackie had taken a different approach to drawing a curve of plaits on the screen, as she was creating new objects for each plait in the curve. Although Jackie was not getting the program results that had originally been demonstrated at the beginning of the session, she was still very engaged in creating her curve on the screen in the manner in which she was able to do so. In addition to creating new plait objects and placing them on the screen using the initial (X,Y) value in the properties panel, she was also making use of the rotate and dilate codelets resulting in an approximation of the results of the initial program demonstration. While I was observing her working, she looked up and asked, "How do you make it do the curve on its own?" There is nothing more gratifying in a high needs school than to have a student say "help me".

I took Jackie back through the original example and demonstrated the “Do While” codelet. I also reviewed the different panels containing the Controls, Methods, and Events. Having completed the review, I did not offer any more suggestions unless Jackie asked additional questions. I paid closer attention to Jackie as she worked for the rest of

the session because I really wanted to know if she succeeded at climbing the learning curve. She continued to work steadily and did succeed at assembling a loop before the session ended.

At some point, I heard what I thought was an “Ah-hah!” from Tomas which immediately drew me back over to where he was working. He had successfully de-bugged his script and discovered that he had put the “Duplicate” codelet in the wrong place. Having moved “Duplicate” to the correct place, the script functioned according to his expectations, which resulted in a very happy Tomas.

In addition to my observations of Tomas debugging a script and Jackie grappling with the beginnings of programming, there were other interesting indications of learning taking place. After hearing a groan from one student, I noticed a hand go up to the screen and trace along the Cartesian coordinate lines of the grid, followed by an "Oh!" and what seemed to be an adjustment of the starting (X,Y) values, terminating in a "Yay!" Another student spent a significant amount of time experimenting with the starting angle of the plait - it seemed as though every time I passed by where this student was working, the plait was being rotated yet again, quite probably through most of the 360 degrees that are available for rotation!

In summary, it seemed that this was a fairly good balance of challenge and ease of use—more or less on target for the ZPD.

5.3.2 Kente Cloth Weaving Simulation with a Class

During the summer of 2012, I had the honor of traveling to Kumasi, Ghana for the second time as one of the graduate student Fellows whose travel was funded by the National Science Foundation GK-12 program. While in Kumasi, we conducted the first usability tests for the Kente cloth pCSDT. The tests were conducted with twenty junior high school students, with the assistance of one of the school's Information and Communications Technology (ICT) instructors. Twenty students working on the software at once would require that they be in groups of two, working on one computer, this was the first field test of the Kente cloth software, and it did not go quite as expected.

The Kente cloth pCSDT was pre-loaded onto the netbooks that the GK-12 grant had provided for our use with these students. There was no indication that there was

anything amiss with the software. The Java virtual machine started when it was selected, the applet loaded, and the interface appeared just as it had on my development machine. In addition, the default script (though very simple) ran without noticeable difficulty. We seemed to be all ready for the students to work with software.

The students assembled at 9:00 am for their ICT class and we passed out the netbooks for the students to use. They started the netbooks and from the windows desktop clicked on the Kente cloth pCSDT. With the interface started and visible, I reviewed the basics of the program with them, briefly, as they had used the Cornrow Curves pCSDT the day before, and therefore, they had some familiarity with this program already. Once we reviewed the basics of the codelets and the panels, the students were encouraged to begin constructing their own Kente cloth designs.

I began circulating from student to student, helping them to get the initial parts of each script constructed properly in the scripting panel. In particular, the first step in the pCSDTs is to find the 'On Begin' codelet in the Events panel and place it in the scripting panel. From there, it is a matter of locating the weave on the screen by the use of 'X' and 'Y' values in the Cartesian plane, and using the duplicate codelet, which creates the weave circles at that location. In moving from student to student through the classroom, it was my experience that roughly half of the students started the script successfully on their own, with the other half needing that initial assistance.

At this point, one of the students called me over and asked me how to construct a row of weaving on the screen. From a pedagogic standpoint, it occurred to me that this would be a perfect time to highlight the ability of the iteration codelets, 'Do While' or 'Repeat' so many number of times. I decided that the 'Repeat' codelet would be a better first introduction to iteration. I reflected the question back to the student to be sure that I had understood her intention correctly "Do you want to create a weaving pattern for a line, without having to create each weave individually?", for which her response was yes. I suggested that she click on the Control tab of the codelet panel, then she did and all of the yellow control codelets appeared. I pointed out the 'Repeat' codelet and she dragged it over and attached it to the 'On Begin' event codelet, which was already present in the scripting panel. This was the first time that I noticed that something was not going well with the software. When the student clicked on the codelet to drag it, it seemed to be

located about two centimeters behind the mouse pointer. The student also misjudged the placement of the 'Repeat' codelet and released the mouse too soon, which caused the 'Repeat' codelet to snap back to the Control codelet panel. What had been a minor 'lag' annoyance with dragging codelets in development, was now manifesting itself as a serious usability issue.

I was aware of the 'lag' that codelets suffered when being dragged from the codelet panel. However, this behavior was not always consistent. On my development machine, dragging was never an issue. However, that machine was far more powerful than the netbooks that were being used by these students. The intermittent behavior in the 'lag' made me think it was just a Java virtual machine error, or perhaps, an issue with this individual computer. However, on the netbooks being used currently, the 'lag' was no longer intermittent, and worse yet, it was causing student frustration like the 'codelet snap back' experienced by this student. Undaunted for the moment, the student again selected the 'Repeat' codelet and dragged it over to the 'OnBegin' codelet in the scripting panel. This time, however, she made certain not to release the mouse button until she was convinced that the codelet would indeed attach itself to the script. We then went on to add the 'Translate 'X" and 'Duplicate' codelet to the interior of the 'Repeat' codelet, and after setting the initial values for the amount of the translation and repeat, the image of a line of weaving successfully appeared on the screen.

During the remaining amount of the lesson time, I circulated and helped to troubleshoot difficulties in programming logic that yielded unexpected results on the screen, however, it became clear that this version of the Kente cloth weaving applet suffered from deep design flaws. This particular user story is recounted and continued as a developer story in section 5.4.1.

5.4 Ethnographic Developer Stories

5.4.1 Kente Cloth

This developer story is a continuation of the user story from section 5.3.2 of this chapter. We pick up the narrative after the lesson has started, when all students have accomplished creating the beginning of a script in the scripting panel, and when all students have started to create more complicated designs with the Kente cloth pCSDT.

Up to this point, the lesson seemed to be going well such that only the 'lag' difficulty in dragging codelets from the Control and Methods panel into the scripting panel being identified as a problem.

The lesson went well initially, however as students began to add additional objects to the program, as they made progressively more complicated designs, it became very clear that the software was not up to the task. The design flaw in the program would take a while to sort out once we returned home, but the results of the design flaw were quickly becoming evident. The more objects the students added to their design, the slower the interface became, eventually grinding nearly to a halt. As it turns out, that first design of the Kente cloth applet drew its weave representation on the screen by drawing Java 2d circles – a lot of them. The computational load in drawing and redrawing of increasingly many circles on each screen update, proved to be too much for the netbooks that we were using. It was definitely an instructive experience for me as a developer. I never noticed any sort of slowdown because my development machine was, at the time, a far more powerful machine than the netbooks. We had finished this version of the applet on our flight to Ghana, and this was the first time it had been used with students.

Worse yet were the side effects, slowing down the user interface. A lag 'bug' that occurred when the user dragged codelets from the methods and control panel into the scripting panel had graduated from minor annoyance to major impediment. As the netbooks were working on redrawing so many circles on the screen to represent the weave, the screen refresh rate plummeted, resulting in what could only be described as a 'hiccup' effect, when the user made changes to the scripting panel. It was clear that this version of the Kente cloth simulation had failed its student users and I called an end to the lesson, apologizing to the students saying that they all had done a great job; however, our software had not properly supported them in their design efforts.

The user experience, detailed above, though difficult for the students and the developer, really helped to move the development of the Kente cloth weaving simulation forward. First, after careful analysis of the applet design, it became clear that using Java 2d circles was not the optimum strategy for this project. In the re-design of the Kente cloth applet, we switched to using OpenGL ellipses to simulate the weave pattern on the screen. Using OpenGL to draw the weave representation resulted in an improvement in

the speed of the drawing over Java 2d and resulted in a more clean and realistic rendering.

Second, as I worked to lower the complexity of drawing the weave pattern, I also recognized another design element that had consequences that I had not noticed previously. The 'game loop' that started once the GUI was on the screen, made its way through the object queue, updating all of the system parameters such as location, color, etc., and then through the Event queue, which handled all of the changes made by the codelets in the scripting panel. Following the object and event queues, the 'game loop' would issue a call to refresh the screen – which in the case of the original version of the Kente cloth applet, meant redrawing all of the weave circles. This happened in every trip through the 'game loop', whether or not the screen needed to be refreshed. This call to 'refresh' was the appropriate design for an animation simulation such as the pCSDT Skateboarder applet, however this was not at all optimal for a drawing applet such as Kente cloth. We added a Boolean value that would allow the 'game loop' to skip the screen refresh if the refresh was not necessary in that trip through the loop.

5.4.2 Adinkra Stamping

The developer ethnographic stories are taken from direct observation, for research that was conducted on site in Ghana and from emails between developers over the development period. The Adinkra stamping pCSDT was an especially challenging applet to develop. This applet sat at the intersection of the cultural simulation that is Adinkra stamping, the mathematics involved in creating the geometric forms found in the craft, and the computational complexity issues that arise from rendering those forms on the screen for the user. We start our story about the Adinkra stamping pCSDT with the interactions with our craftsperson, Gabriel Boakye, in Ntonso Ghana.

The initial interviews that were conducted with Gabriel took place in the summer of 2011 and are covered in the Software Development chapter of this work. We pick up the story with the events of our return visit to Ntonso in the summer of 2012. We worked on the Adinkra stamping tool from the time of our initial interview in the summer of 2011 to our return in the summer of 2012. We presented software to Gabriel at his shop in Ntonso. In our second encounter, we gathered additional feedback that indicated that

although he was pleased with our work, we did not have everything exactly correct in the simulation.

Frequently, there are tradeoffs that are required as we balance fidelity of the craft production in the real world with our pedagogical renderings in the virtual world. In this case, we had placed the Cartesian grid that serves as scaffolding to the user, to aid in design element placement, such that all four quadrants were available to the user. Gabriel immediately determined that from a craft point of view, an Adinkra artisan would never look at stamp placement in that way. This was a particularly interesting discussion because we had made that design decision based on my desire to include the four quadrants from a mathematics pedagogic point of view. In the end, we deferred to Gabriel and placed the entire simulation output screen into the first quadrant with coordinates running from 0 on the left increasing to the right and vertically.

Development continued on the Adinkra stamping pCSDT throughout the next year. The new goal was to solve all of the issues with the software in order to test the software with the students at the junior high school in Kumasi, in the summer of 2013. To accomplish that task, several issues needed to be resolved in developing the application. The developers needed to decide on how the software should treat the starting angle for spirals. The spirals drawn to the screen needed to be scaled and matched to a spiral drawn in the real world, on graph paper – using the same parameters for both physical and virtual representations. For this to work, in particular, for junior high school students, the software would need to take parameters measured in degrees, convert them into radians for the figure that would be displayed on the screen and back again as output for the student user. The development effort would need to be completed before departure, which was scheduled for the 1st of July.

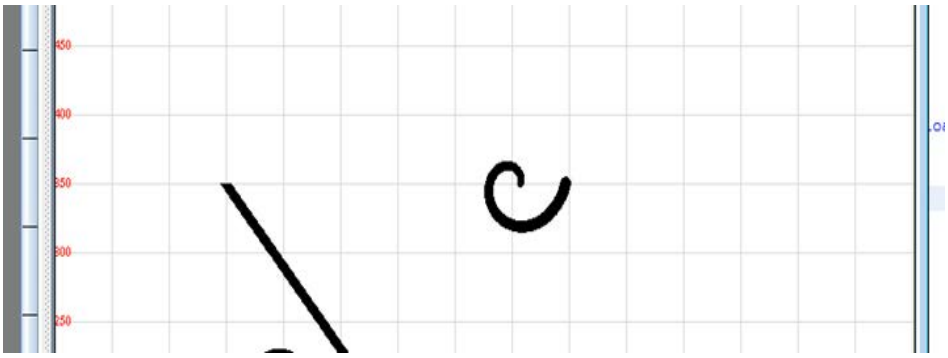


Figure 5.1: Adinkra stamping pCSDT logarithmic spiral example

There was great difficulty in deciding how to handle the 'start angle' of a spiral. There were two ways of thinking about the start angle, one is the math way and the second was the 'turtle' way [79]. The math way would orient the angle direction as the degree. Note in the figure above, that would seem to be a start angle of 90 degrees as the figure (from the innermost part of the coil) starts by heading 'north', or parallel to the positive 'Y' axis of the Cartesian plane. The 'turtle' way is an inheritance from the Logo software tradition, which probably drew it from mapping: imagine that you are a 'turtle', walking 'north' (parallel to the positive 'Y' axis) – since this is the regular way of walking (walking straight ahead) then that should actually be at a starting angle of 0. This difference represented a total of 90 degrees out of phase from the 'math' way. Which, we can ask, is the right method, 'math', or 'turtle', to model this situation? Which is more useful in the classroom, the 'math' way or the 'turtle' way?

It was very challenging to get all the parameters involved in simulating a logarithmic spiral to correspond in a sufficiently convincing way, with a spiral one might draw on a piece of paper. The parameters involved are not just those found in the equation $r = a^{b\theta}$. When creating a spiral on the screen, there are the extraneous parameters such as openGL grid size for the output screen. The Cartesian plane background lines (or image) that is totally unrelated to the spiral figure, but needs to be sized appropriately to match the drawn spiral, which to be meaningful as a pedagogic tool, needs to match physical graph paper used by the student.

It was tricky to handle spiral attenuation, as the spiral arm gets larger. The spiral is actually drawn as a series of circles along the path that the spiral sweeps out as it grows. There is a point in the growth of the spiral where the fixed circle size begins to

separate, resulting in a dotted line rather than the smooth curve as appears above. As the arc begins to sweep out larger distances for each degree measure in the arc, the spiral needs to have additional circles added to the arc to undo the dotted line affect.

In addition, the calculation for the spiral needed to be handled in degrees. The students using the applet would likely be in grades 6 through 8 and would not have encountered trigonometry in their math classes at that grade level. Thus, the applet needed to accept measurements in degrees, which the students would be likely to understand. However, all the calculations in the back end generating the figure on the screen are done in radians. This added a layer of complexity in translating degrees entered by students to radians, and then translating radians back to degrees to display to students. Further, all of this needed to be scaled to the screen appropriately to match the protractor that a student was likely to hold up to the image to measure the degrees from the onscreen image.

As an example of the difficulty:

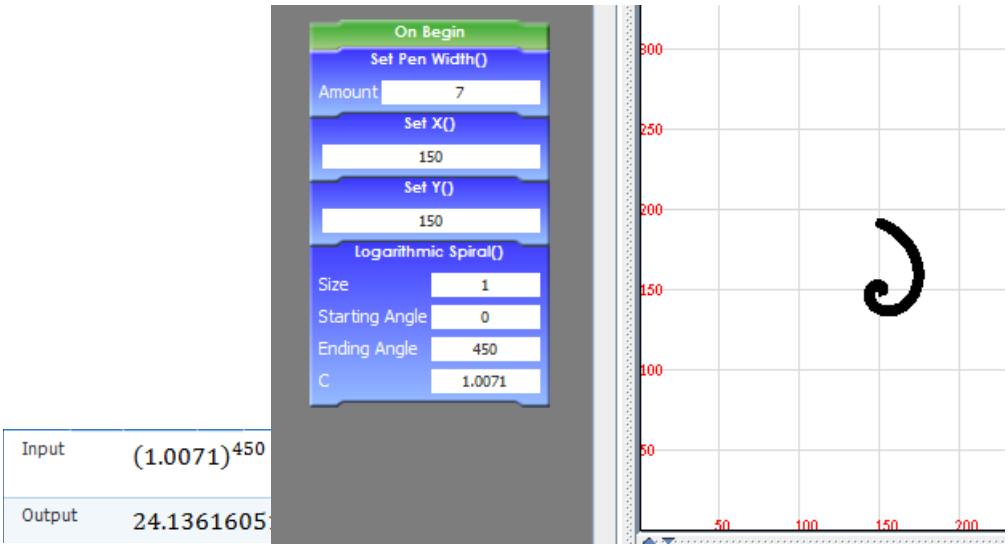


Figure 5.2: Adinkra stamping pCSDT screenshot

The goal in this case was to make a table with the radius for angles every 90 degrees. For the angle value 450 degrees, the radius equals about 44, which from Figure 5.2 above is (about) 193 - 150 = 44 or so. However, the equation yielded an answer of 24.

As we see in the above description, the final design is a compromise between not only the demands placed upon it by cultural, pedagogical, and user (HCI) forces, but also the computing environment.

5.5 Conclusion

Ethnographic studies of both users and developers have proven invaluable in the pCSDT software development effort. In this chapter, we have reviewed these stories for user interactions with the Cornrow Curves and Kente cloth applets. The user experience, as disastrous as it was in the initial rollout of the Kente cloth application, resulted in an informed approach in the applet re-design that resulted in a greatly improved program. It is also clear that changes identified in working with the Kente applet also benefited the Cornrow Curves applet once implemented.

We also looked at the developer experiences for the Kente cloth and Adinkra stamping applets. Our examination of the challenges in developing the Adinkra stamping applet demonstrated how the different aspects under consideration, in this case, the user experience, the mathematics of the curves, and the computer science of rendering the image on the screen, all needed to find common ground. This ‘sweet spot’, at the intersection of user experience, math, and computer science is often a difficult place to identify and build. Nevertheless, when we do succeed at locating our software at just the right spot, we end up with a pedagogically valuable tool that succeeds at teaching both the math and computer science concepts involved.

6. SUMMATIVE DATA ANALYSIS

6.1 Introduction

In this chapter, we review the results of an experiment that we conducted during the summer of 2014 at a Junior High school, in Kumasi Ghana. Pretest and posttest data for this experiment was collected under a protocol approved by the Rensselaer Institutional Review Board (#998), and with the consent of all parents/guardians, the consent of the Ayeduase Junior high School, and the assent of all students involved in this study. As discussed in chapter 1, this summative evaluation is used as a final check on the viability of the software in meeting our goals of improving the math and computing performance and interest of students, in this case "postcolonial" students in West Africa.

6.2 Ghanaian Adinkra Symbols and Logarithmic Spirals

Adinkra symbols can be primarily observed today in Ghanaian textiles. The Akan peoples of Ghana adopted Adinkra textiles around the year 1800, yet the origins of the craft remain uncertain [51]. Many of their geometric forms exist in older archaeological artifacts, across a wider geographic range. In the case of the textiles, these were originally used in the funerary arts with each symbol communicating a particular idea to the departed loved one. Contemporary uses of Adinkra symbols have expanded well beyond the funerary arts. Traditional Adinkra artisans in Ntonso, Ghana still carve symbols from the calabash gourd, make their own ink, and stamp various types and styles of cloth; primarily for tourists that visit their shops. A drive through nearby Kumasi reveals Adinkra symbols painted on garden walls, the columns of Internet cafes, and molded into the backs of plastic chairs. In the 21st century, Adinkra has become a global phenomenon. In the United States, Adinkra symbols adorn everything from t-shirts and jewelry to braiding salons, and their forms and names are used in lively and creative ways by African American community organizations and hip-hop artists.

Portions of this chapter have been submitted to: W. Babbitt, M. Lachney, E. Bulley, and R. Eglash, "Adinkra mathematics: a randomized, controlled study of ethnocomputing in Ghana," *For the Learning of Math.,* submitted for publication.

During the time period 2010-2014 we engaged in ongoing ethnographic research on the mathematical and computational significance of Ghanaian Adinkra symbols. This research included teaching interventions in Ghanaian junior high schools. The foundation of this work is based on Eglash's [45] research that documents Ghanaian pre-colonial knowledge of logarithmic curves in symbolic representations of organic growth. Western mathematicians have long recognized logarithmic curves as a defining characteristic of organic growth. Darcy Thompson's 1917 classic, *On Growth and Form,* was one of the first works to provide a formal analysis. Today specific examples such as the Fibonacci sequence in plant spirals have become a math textbook staple, while more complex theories for the ubiquity of power laws in biological morphogenesis are the subject of significant research programs [127]. While we do not want to attribute understandings that are not actually present, there is solid evidence that pre-colonial Ghanaian designers consciously employed logarithmic scaling--in particular the log spiral--as a visual model for the underlying geometric forms common to living organisms. Adinkra symbols do not just mimic organic growth; they are a means of representing a hybrid knowledge form at the intersections of biological, mathematical, and social concepts.

Figure 6.1: Dwennimen

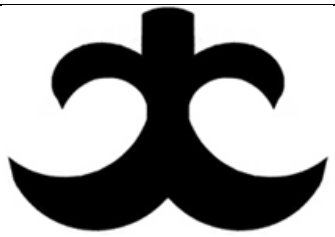


Figure 6.2: Akoko nan

Figure 6.3: Sankofa

In the pre-colonial Ghanaian context, the logarithmic curves found in Adinkra designs are consistently associated with biological structures. Examples (Figure 6.1-3) include the ram's horn, chicken's foot, and curve of a long-necked bird. Each symbol represents a colloquial saying connected to cultural and ethical values. For example, Dwennimen, the ram's horns, is associated with the saying "it is the heart, and not the horns that leads a ram to bully [51]". It is not genetics (the horns you were born with), but rather your efforts (from your heart) that matter.

Figure 6.4 shows a fourth Adinkra symbol that uses log curves, “Gye Nyame”. It has a stronger mathematical significance: while the other symbols show log curves associated with a particular biological structure, the Gye Nyame symbol is a generalization of log curves as emblematic of life in general. The saying associated with this symbol is “no one except for God”. The bumps down the center represent the knuckles of a fist; a symbol of power. At each end there is a logarithmic curve, the curves of life [128]. Thus, the aphorism becomes less cryptic: “no one except God holds the power of life.”

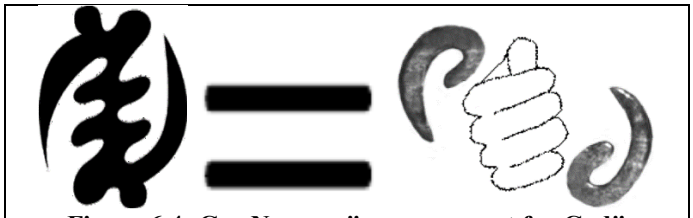


Figure 6.4: Gye Nyame: "no one except for God"

This syncretic mathematical/cultural/biological significance of logarithmic curves in Adinkra forms the basis for our educational interventions. The logarithmic curves of Adinkra not only vary across symbols but also in different variations of the same symbol, creating a rich body of geometric forms suitable for discovery or inquiry learning. Are certain symbol curves quantitatively similar to their biological sources of inspiration? How does mathematics model the distinctions our eyes and visual intuition tends to make? Does the variation that different artists give to the same symbol indicate differences in skill, media, traditions, or other affiliations? Because the 2D form requires a more complicated description, we began by considering only the 1D “edge” of these shapes, modeling them as the arc of a logarithmic spiral. Thus we could focus on two parameters: the angle “sweep” of the log spiral arc, and a constant C that determines the overall shape (from the equation in polar coordinates: $Radius = C^{\theta}$). We referred to the constant “C” as the amount of “coilness” (either tightly or loosely) in the curve, in our work with the Ghanaian junior high school students. For example, consider the Adinkra symbol Sankofa, which means “you can always return to your roots” (hence the bird looking backwards). In Figure 6.5a, we can see that the logarithmic spiral makes up the curve of the Sankofa’s neck. Variations to the design of Sankofa result from the changes

in the exponential parameter or coilness of the neck. Figure 6.5b has a smaller exponential parameter than Figure 6.5c, which results in the more closed, tightly coiled spiral. Figure 6.5c has a larger exponential parameter resulting in a more open, loosely coiled spiral. As we describe in the next section, this mathematical insight was designed into the log spiral "block" of Adinkra Computing CSnap software as the value "C".

Figure 6.5: Logarithmic curve and Sankofa, left; Sankofa symbol with different coilness, middle and right

Adding the mathematical significance of Gye Nyame, Sankofa, and other Adinkra symbols to Ghanaian educational contexts could create a valuable alternative to dominant curricular models, which simultaneously claim to be non-cultural abstract universals, and at the same time make clearly Western references (Pythagorean Theorem, Archimedean spiral, etc.). Paulo Freire [113] argues that the decontextualization of education from learners' concrete experiences is alienating. While the Ghanaian national curriculum has made an admirable effort to include Adinkra and other local cultural resources in its humanities curriculum, Freire's critique is still applicable in the case of math and science in Ghanaian schools.

The study of Ghanaian culture could be incorporated in schools starting in the lower primary levels where students are taught how to identify and draw geometric shapes. Pupils could draw some of the easiest traditional symbols, such as the Akoma (heart shape), in addition to other basic shapes already in the curriculum. This would make it easy for students to view their heritage as having contemporary significance rather than merely a holdover from earlier times. However, this cultural background is more often taught as part of the Ghanaian language subject in junior high schools; there is no connection to the study of math, science and technology. Despite the clear presence of mathematical concepts such as geometric transformations, the Cartesian plane and basic

computations employed by artisans in the making of the Adinkra symbols, none of these resources are fused into the teaching of math and science in the classroom. Most of the examples and illustrations given to students are purely abstract theories, with any concrete illustrations taken from the most generic examples.

One issue that challenges the fusion of culture into math, science and technology education is the fact that each of the ten regions in Ghana has differences in their cultural practices and values. In effect, there are different contents for the Ghanaian language curriculum and syllabus depending on the region a student finds him or herself. For instance, while a student in the Northern Region may be learning about mud architecture, those in Upper West region might be learning about the Xylophone as part of their cultural heritage. This is another reason why Adinkra is particularly appropriate; the symbols have taken on a status as part of the Ghana national culture shared by all regions, and even internationally as symbolic of African heritage in the black diaspora.

6.3 Experiment Overview

Our teacher collaborator, Enoch Bulley, selected 20 students from his ICT classes at the Ayeduase Junior High School. He chose 10 students from the seventh grade and 10 students from the eighth grade. These 20 students were then assigned randomly, using a random number table to two groups. We have designated these two groups as the control group and the intervention group.

This experiment tested the effectiveness of teaching the mathematics of logarithmic spirals in a "quasi-intervention evaluation" [128]. We refer to the experiment that we conducted as "quasi-experimental" because, as stated in the literature, we could not control every variable that we encountered outside of a laboratory setting. However, we did control what we felt were the most important variables as all students were from the same school, drawn from the 7th, and 8th grades, with the lessons taught by the same teachers.

The control group lesson occurred prior to the intervention group lesson. Both interventions began with a pretest and ended with a posttest that measured the student's knowledge of the topics covered in the lesson. The control group received two days of

instruction using the lecture and GeoGebra software simulation. The intervention group received three days of instruction using lecture and Adinkra Computing in CSnap. The extra day in the intervention group intervention was included for students to explore the cultural significance of Adinkra. Instruction time on the math of logarithmic spiral was equal in both groups.

6.4 Control Group Lesson

The mathematics of logarithmic spirals was taught to the students in a similar format of lecture, followed by reinforcement of concepts taught through the use of simulation software. The control group lesson was developed using a freely available website that details the mathematics of logarithmic spirals. The control group lesson used a GeoGebra based logarithmic spiral applet to reinforce the mathematics lesson based on the website. In addition, the control group lesson also included a teacher led guided practice using the GeoGebra software, as well as a group practice where students collaborated on a learning project.

6.4.1 Class Period Overview

Class Period 1 Overview:

- Administer pretest for logarithmic spirals (Appendix 1).
- Begin reading from the instructional handout
- Develop Cartesian plane at the chalkboard
- Define exponential parameter
- Using the unit circle at the chalkboard, develop angles and degrees
- Open GeoGebra and load the log spiral applet
- Develop graph of e^x on the chalkboard
- Develop the graph of $y = x$ and $y = \ln x$ on the chalkboard
- Develop what we mean by exponential growth
- Continue reading from handout

    - Log spiral approximations
    - Dividing a golden rectangle into squares yields a logarithmic spiral
- Develop tangent vector
- Return to GeoGebra and point out the tangent vector in the log spiral applet.

Class Period 2 Overview:

- Review of previous days topics
- Continue reading from the instructional handout, spirals in nature.
- Group activity using GeoGebra, matching exponential parameter to log spiral background graphics.
- Administer posttest.

6.4.2 Class Period Narrative

The lesson started with a lecture with one of the authors at the chalkboard at the front of the class, with the students seated up front with their copies of the handout. Many students also had paper, pens and pencils to take additional notes. The schedule for the first day of our work with the control group (see section 6.4.1 above) started with the administration of a pretest, then students listening to a lecture while following along using the printed handout, and then finally students reinforcing what they learned through the use of the GeoGebra computer simulation for logarithmic spirals.

The lesson plan for the lecture portion of the learning experience involved one of the authors standing at the chalkboard drawing and explaining the necessary concepts for the lesson, while making this activity as interactive as possible. This interaction was sometimes difficult as the students were not always willing to share what they knew with the author. The lecture included the following concepts, working up from the ground floor, assuming little or no prior mathematical knowledge. We began with a review of the Cartesian plane, 'X' and 'Y' axes, coordinate pair notation, angles, measuring angles in degrees, the unit circle, rays, radius, and exponents, followed by the introduction of the spiral formula $r = be^{ax}$.

Next the notion that alpha represents the exponential parameter for the logarithmic spiral was introduced showing that as alpha changes, the spiral 'opens up' or 'closes'. We introduced the graph of $y = e^x$, drawing it on the chalkboard and we discussed the concept of exponential growth, showing that small changes in the X variable result in large changes in Y. Finally, the notion of tangent vector was developed on the board, both on the unit circle and on the logarithmic spiral at the chalkboard.

With the completion of the lecture, the students returned to their desks along the outer walls of the room. For the next hour or so, the students worked with a logarithmic curve applet loaded into the software called GeoGebra. This applet allowed students to adjust the parameters for the formula $r = be^{a\theta}$ and see the effects of these changes in the spiral drawn on their screens.

The authors circulated about the room encouraging students to interact with the software, making sure that each student succeeded at adjusting the parameters 'a' and 'b', as well as having each verbalize what happened to the spiral with each parameter change. The exchanges usually ran something along these lines:

Author: Did you change the values for ‘a’ and ‘b’?

Student: Yes.

Author: Tell me what happened to the spiral.

Some students reported that they had changed the values but were unable to verbalize a description of the resulting changes to the spiral on the screen. When this occurred, the author worked with the student changing the values to demonstrate the spiral opening and closing on changes to variable 'a', and rotation on changes to coefficient 'b'. This particular GeoGebra applet also included a tangent line simulation represented in the applet as 'speed'.

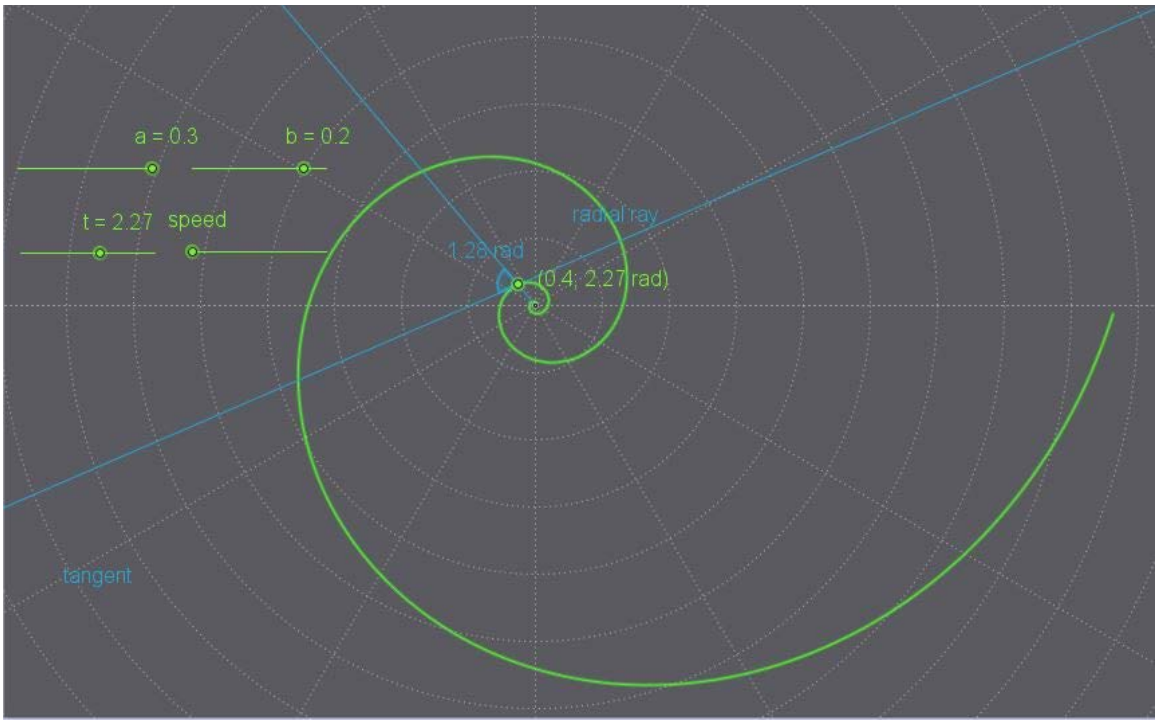


Figure 6.6: GeoGebra log spiral application

On the second day of instruction, the review of the previous day's work was accomplished by one of the authors standing at the chalkboard guiding the review, but at each step, asking the students to identify or otherwise talk about each of the topics that they worked with. Again, starting from the ground up of the Cartesian plane, the students were asked as a group to identify each concept.

At the completion of the review, the group returned to the handout to look at pictures of logarithmic spirals in nature. Such spirals included the shells of snails, the ends of pinecones and the seed arrangement in sunflowers. Following this discussion of logarithmic spirals in nature, the students returned to their desks to once again work with GeoGebra. The spirals in nature applet for GeoGebra placed a logarithmic spiral on a background that showed items from nature, for example the swirl of a hurricane, water droplets coming off of a wet spinning tennis ball, snail shell and sunflower. This started as an independent activity and then transitioned into a group activity with students huddled around a single computer with the display being projected on the wall. Student interaction began very quietly, but as time went on and the students became more comfortable with working together, they began shout out different values for the

coefficient 'b' and exponent 'a' that they thought would make the spiral match the background. At the completion of the activity, the students had managed to determine values for 'b' and 'a' that convincingly matched the drawn spiral to the background graphic in the applet.

At the completion of this activity, the students completed the posttest and were dismissed from the classroom.

6.5 Intervention Group Lesson

6.5.1 Class Period Overview

Class Period 1 Overview:

- Administer pretest for logarithmic spirals (Appendix 1).
- Motivate discussion by asking students about Adinkra.
- Develop Cartesian plane by comparing to Adinkra Kronti Ne Akwamu Adinkra symbol.
- Introduce and start CSnap application
    - Import Akoma script.
    - Clear Akoma script.
    - Add Cartesian plane costume to the stage.
- Introduce Logarithmic Spirals using student handout, focusing on the Adinkra Dwennimen symbol.
- Guided Practice:
    - Return to CSnap
    - Clear Akoma Script
    - Rebuild the Akoma Script with students, step by step, explaining each code block.
    - Provide logarithmic spiral parameters for the left half of Akoma. Students to complete the right half on their own.

Class Period 2 Overview:

- Introduce logarithmic spirals vs. linear spirals.
- Introduce natural versus man made spirals.
- Review curves in nature from the student handout.
- Review logarithmic curves present in Adinkra symbols.
    - Dwennimen: the ram's horns
    - Sankofa: "return and get it."
    - Akoko Nan: chicken's foot.
    - Gye Nyame: "Except God."
- Introduce concept of exponential growth using powers of 2 and cell division.
- Design challenge:
    - Open 'confused' Dwennimen (Figure 6.13).
    - Challenge students to complete Dwennimen so that the spirals match the goal image.

Class Period 3 Overview:

- Review all key concepts using Adinkra symbols as examples
- Introduce Tangent Vectors using the student handout, continue with CSnap:
    - Open CSnap.
    - Load the incomplete Mpuannum project.
    - Encourage students to complete Mpuannum (Figure 6.15) challenge.
    - Administer posttest.

6.5.2 Class Period Narrative

The intervention group lesson used the CSDT CSnap Adinkra Computing applet to reinforce the mathematics lesson based on the Adinkra symbols. Like the control

group, the intervention group also included a guided practice and a group collaboration project. The intervention group met for 3 days, one day longer than the control group. This group had very little lecture at the chalkboard, and when there was work at the board it was always framed in the context of Adinkra symbols. For example, an Adinkra symbol was used to motivate a review of the Cartesian plane and when logarithmic spirals are introduced, they too are done within the context of an Adinkra symbol. Because the experimental group had the added cognitive load of learning about Adinkra and the CSnap programming environment, they received the intervention for an additional day.

The intervention lesson included the same concepts as those used with the control group, however each of the concepts was motivated and taught using an Adinkra symbol. For example, in a review of the Cartesian plane, we used the 'Kronti Ne Akwamu' (Figure 6, the symbol for "the dual nature of life and democratic decision making in the State of Ghana [51]"), which enabled us to draw out the comparisons between that symbol and the Cartesian plane. This enabled us to expose the embedded mathematical structures present in the symbol to the students. The intervention group started with the pretest on the first day. Once finished, they gathered at the front of the classroom to begin review of the student handout.

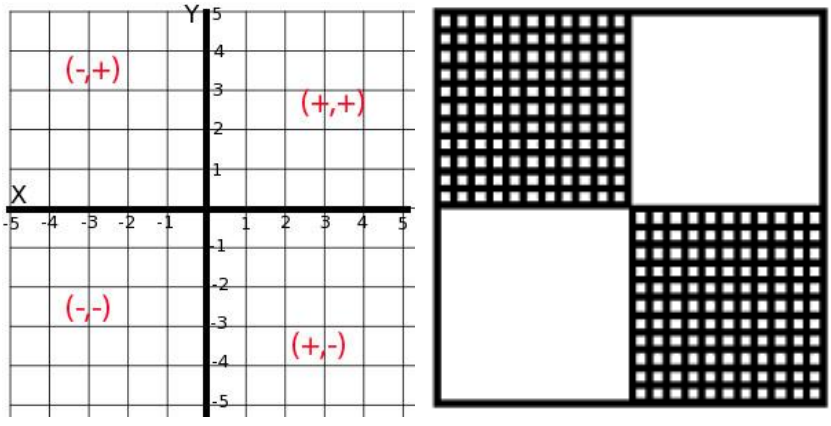


Figure 6.7: The Cartesian plane, left; Kronti Ne Akwamu Adinkra symbol, right.

In using this symbol to review the Cartesian Plane, we see that where two coordinate numbers are the same sign, say (1, 1) or (-1, -1) we see that the symbol is void as there are no lines in those two quadrants. The lines are drawn, in a sense, where people live. In a democracy, people live in spaces that involve having differing opinions and in the case of Kronti Ne Akwamu, that is where the signs of the numbers are different like (-1, 1) and (1, -1). Following the review of the Cartesian plane using Kronti Ne Akwamu, we began our work in CSnap.

The students opened the CSnap environment, imported the Akoma project and added the Cartesian plane graphic to the background of the output window. During this process we oriented the students within the CSnap programming environment. This part of the activity is not without difficulty as there is quite a learning curve to accomplish in mastering the mechanics of loading and saving projects, where to find appropriate code blocks, how to drag them into the scripting panel, and how to 'run' the script in the programming interface.

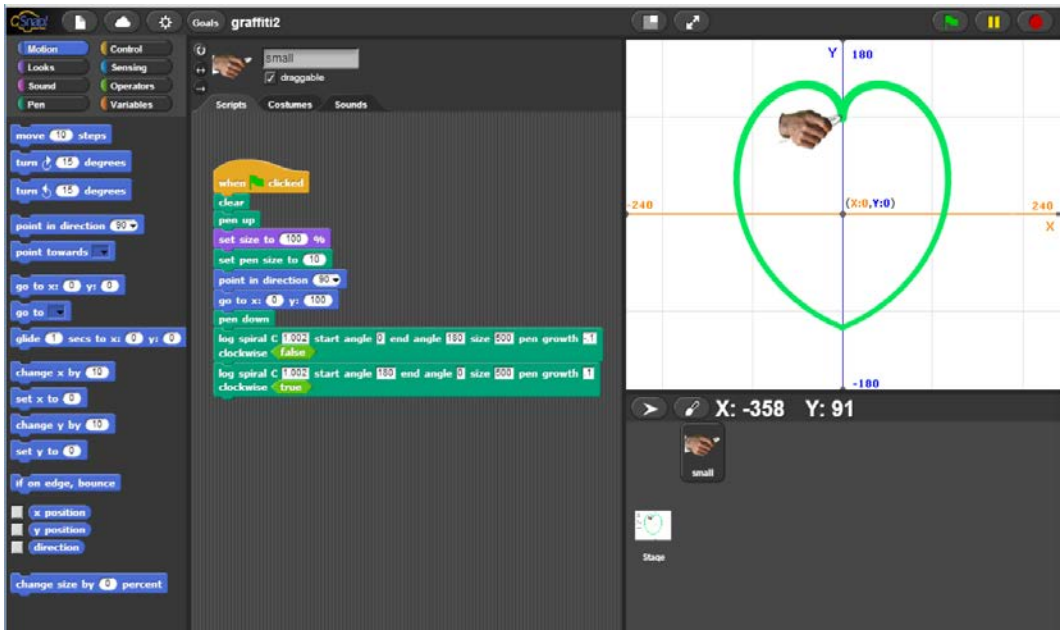


Figure 6.8: CSnap interface with Akoma script running

In Figure 6.6 we see the simulation for the Akoma Adinkra symbol, which stands for “love, good will, and patience [51]”. The user interface shows the programming building blocks in blue (left most column), the scripting panel (middle column), and the

output window on the right. The user (student) designs an Adinkra symbol by dragging code blocks (i.e. black-boxed rules and functions) from the leftmost panel into the middle-scripting window. Students typically arrange the code blocks using some combination of planning and trial and error experimentation, checking each time the script is run to see if the results are close to the desired design.

For those curious about the internal operation of the code, a good analogy might be a script for a play in which each actor reads his or her part. When the user presses the green “play” button, she puts the applet into the running state, which triggers an event queue to cycle through all of the system objects (the actors) in the queue. This updates the system values for all of the object attributes affected by the code blocks (the script) that have been added to the event. The play button is one of these events, but a code block can send out triggers to other code blocks. The updates to values and any other changes to system parameters result in the alteration of the behaviors (typically graphical) that appear in the output portion of the user interface.

Unlike similar programs such as Scratch, these code blocks offer operations specific to many of the Adinkra designs, and their execution is as close as possible to the original Adinkra artisan practices. For example, at the end of the script in Figure 8, two log spiral blocks make up the curves of Akoma. The “costume” worn by the object leaving these curved paths is a photo of the hand of master carver Paul Boakye, holding his carving knife. We took pains to use an algorithm that always orients the blade along the tangent to the curve; thus adding both cultural accuracy and mathematical learning content.

Following their work with Akoma, the students returned to the handout and logarithmic spirals were introduced through the Adinkra symbol called Dwennimen. The Dwennimen Adinkra symbol is composed of four spirals and is frequently compared to ram's horns. The spirals are in pairs because this represents two rams butting heads. "It is the heart, and not the horns, that leads a ram to bully [51]”. The exponential parameter was then introduced as what controls 'coil-ness'. The smaller the exponential parameter the more tightly coiled or closed the logarithmic spiral. The larger the exponential parameter the more loosely coiled the spiral.

Figure 6.9: Dwennimen Adinkra symbol

The students returned to working in CSnap and a more thorough review of the interface was conducted. Code blocks that control clearing the screen, raising and lowering the pen for drawing on the screen, setting the size of a costume graphic, setting the size of the pen for drawing, changing the drawing direction, and the log spiral code block were introduced and explored. At this point in the intervention, the amount of time actually spent working with the mathematics of the logarithmic spiral has been about twenty minutes. This time came through recreating the Akoma shape which required adjustments mostly to the exponential parameter C in the log spiral code block, starting and ending angles and pen growth.

The second day of the intervention began with a discussion of logarithmic spirals versus linear spirals. Logarithmic spirals are found more in nature where linear spirals are more likely man made. Following along with the handout, the idea that snails, aloe plants, the Hawaiian fern etc. all exhibit the shape of the logarithmic spiral. Following the examples from nature, we reviewed the Adinkra symbols that use logarithmic spirals such as Dwennimmen (ram's horns), Sankofa which means "return and get it", Akoko Nan which is the chicken's foot, and Gye Nyame which means 'Except God.'

Figure 6.10: Dwennimen

Figure 6.11: Akoko nan

Figure 6.12: Sankofa

The author then developed the concept of exponents and exponential growth at the chalkboard. Following this work at the board, we returned to working in CSnap and the second of our design challenges.

Our work today was then reinforced through working in CSnap on the Dwennimmen challenge. When the Dwennimmen challenge project is loaded into the software, it consists of 3 'confused' logarithmic spirals. To complete the challenge, students need to determine the correct parameters in the logarithmic code blocks that will make the spirals uniform in size and shape, in addition to adding the fourth spiral.

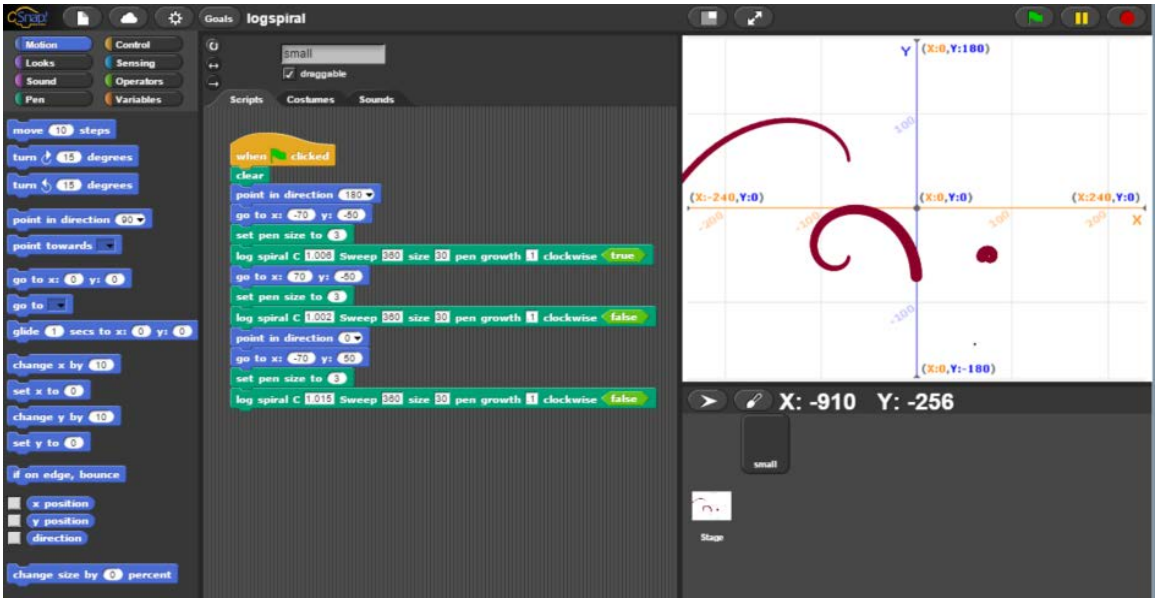


Figure 6.13: CSnap interface with the 'Confused Dwennimen' design challenge

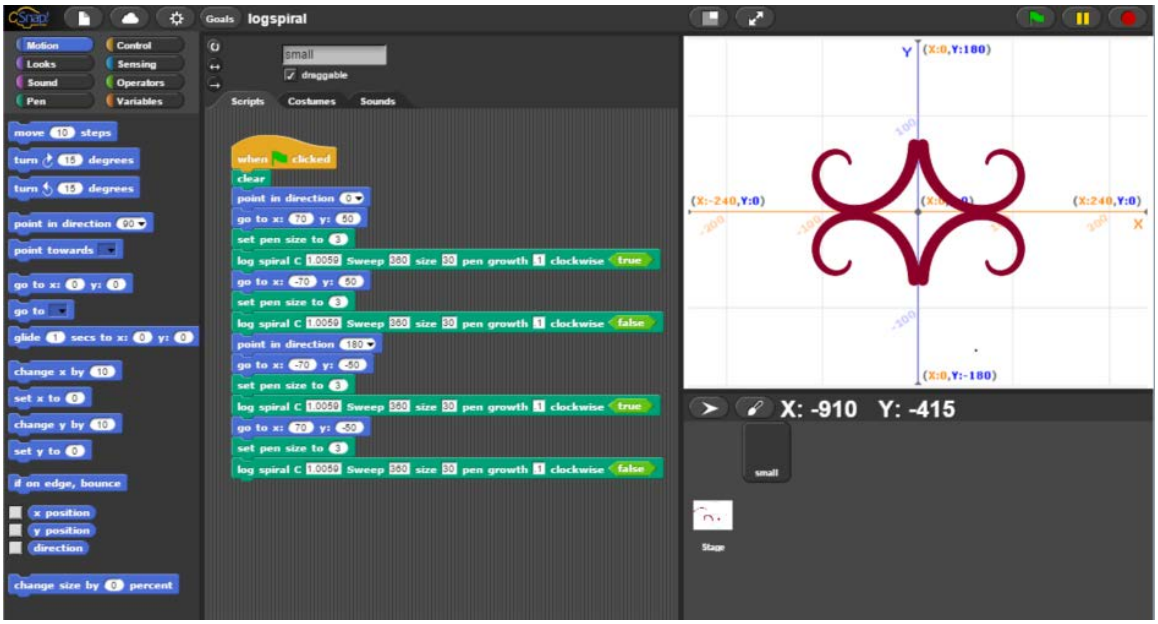


Figure 6.14: CSnap interface with the Dwennimen challenge, completed

The third day of the intervention started with the students all sitting up at the front of the class. We began the lesson by introducing the concept of tangent vector on the chalkboard. This was followed by reinforcing the concept of tangent vector by using the Mpuannum Adinkra symbol in the CSnap software. The Mpuannum design challenge provided students with the first three circles of Mpuannum as shown in Figure 6.11. The student's task was to add the two remaining circles tangent to the center circle as shown in Figure 6.12.

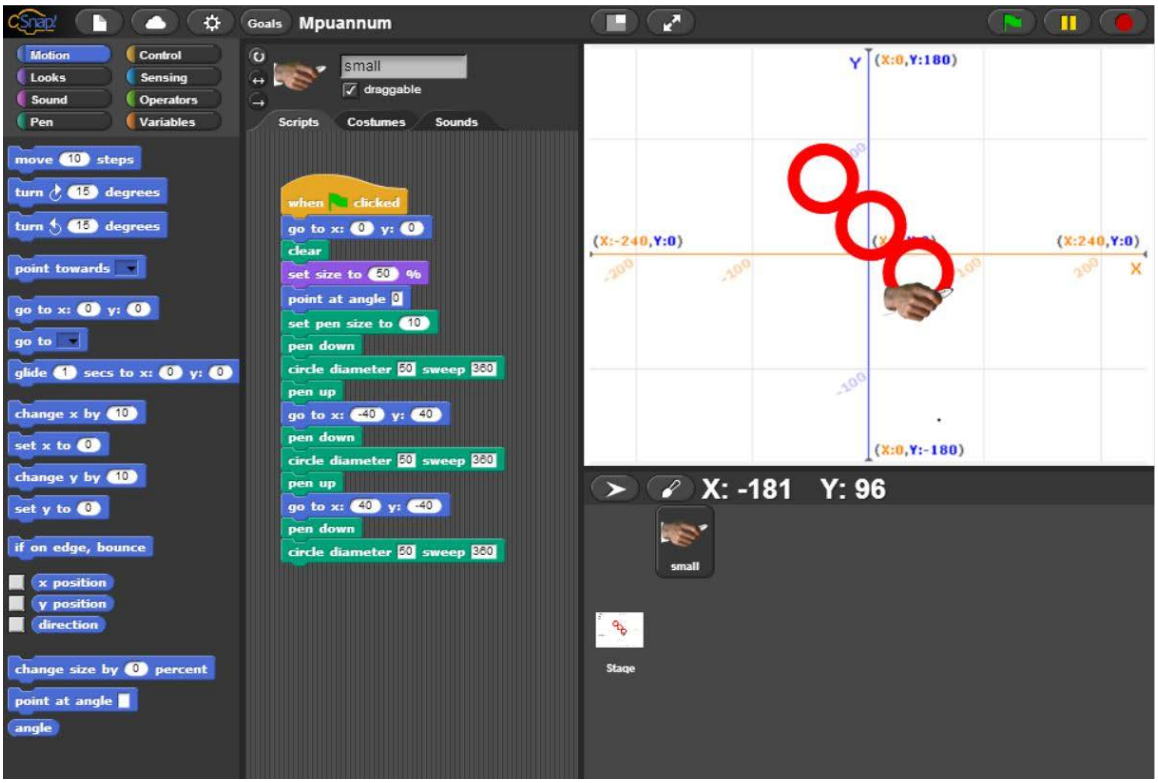


Figure 6.15: CSnap interface with the Mpuannum challenge

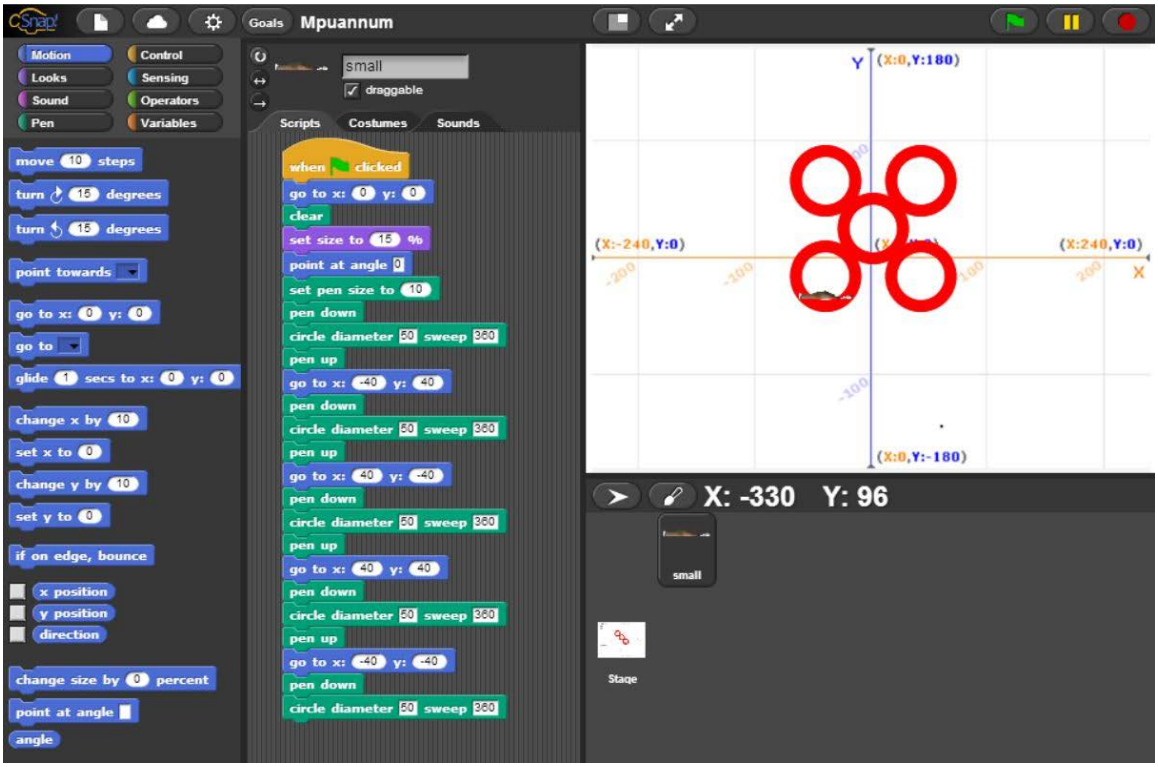


Figure 6.16: CSnap interface with the Mpuannum challenge, completed

One of the interesting things that became apparent as the authors observed the students working on the Mpuannum challenge was their difficulty in handling the 'pen up' and 'pen down' commands so as to draw circles but not have lines between the circles. The proper sequence should be 'pen up', go to (X, Y), 'pen down', and draw the circle. The placement of both 'pen up' and 'pen down' proved quite challenging to most students.

Following the completion of the Mpuannum challenge, students returned to their seats around the outside of the room to complete their posttests. As they completed their posttests, many of the students handed the test in before time was called and they each requested to be able to resume working in CSnap. This was a very interesting result as clearly working with the software created a motivation to continue working even when they did not have to.

6.6 Results

The pretest and posttests formed the independent samples for a t-test. The results showed a significant advantage for the scores for the Adinkra computing based lesson (M = 45.22, SD = 18.67) in comparison to the GeoGebra computing based lesson (M = 13.87, SD = 15.93); the difference was statistically significant at the .001 confidence level.

Although we believe that the use of the Adinkra Computing CSDT to reinforce the learning in the intervention group was a success, we realize that there are limitations in this particular quasi-experiment. One possible limitation is that our sample size is small with only 19 student's total (10 students for the control group and 9 in the intervention group). This number however, was as large as possible, given the constraints placed on us by the number of computers available for student use. We wanted to make sure that each student had access to their own machine to limit the potential effects of students having to share equipment. The students used all available computing equipment in the classroom, including all teacher-machines.

In future research we hope to examine the role that heritage variation might play in these results. In this case, we carried out the testing in the Akan cultural region; thus, many of the students would likely have been of Akan extraction. Since the Adinkra symbol set belongs to the Akan people, we may see different results elsewhere in Ghana.

However we have seen a great deal of variation with CSDTs in the US, with some African American students expressing more interest in Native American design tools and vice-versa [21], [120].

One of the most exciting outcomes of the experiment was the reaction of the intervention group students after the intervention was over and the posttest completed. Unlike the control group students who immediately wanted to leave the classroom, the intervention group students reclaimed their computers and formed small groups at the front of the room to continue working with the Adinkra Computing CSDT. This is not unique to Ghana; similar outcomes have been observed in the US for African American, Latino and Native American students; a promising sign for this approach.

6.7 Conclusion

In this chapter, we reviewed the results of an experiment that we conducted during the summer of 2014 at a Junior High school, in Kumasi Ghana. As discussed in chapter 1, this summative evaluation is used as a final check on the viability of the software in meeting our goals of improving the math and computing performance and interest of students, in this case "postcolonial" students in West Africa. The results of the experiment showed that the intervention students outperformed the control group students yielding a statistically significant outcome as shown by the pre and posttest data. Thus we are able to reject the null hypothesis that using the pCSDT CSnap had no effect, and accept the hypothesis that the use of the CSnap pCSDT did have an effect in the outcome with these students.

7. CONCLUSION

7.1 Contributions to Science

Research Question 1: How do we model the formal systems that underlie cultural deigns such as textiles, artistic practices, and adornment?

Modeling the knowledge systems that underlie cultural artifacts and practices (e.g., textiles, artistic practices, and adornment) cannot be accomplished simply by analysis of a design as an outsider. The elicitations of the software requirements for the Kente Cloth and the Adinkra stamping simulations are clear examples of an inquiry approach that goes beyond observation and analysis to include a research paradigm that is informed by the particular mental processes of the researched. Our Kente Cloth weaving expert clearly held all of his knowledge about his craft 'in his head' as he put it. It took a great deal of working together, but with time and effort, a mental model emerged, for me, of his perceptions of how he practiced weaving Kente Cloth. Our Adinkra expert, likewise, maintained a great deal of his craft 'in his head'. However, when comparing the mental models used for each particular craft, a difference in the complexity of each mental model emerged. The Kente Cloth weaver maintained many different patterns, requiring recall for each to the extent that he did not need to measure or count in order to produce his cloth. This sustained mental effort, I think, is what sets the two models apart. Arriving at this conclusion required careful questioning, and even more so, careful listening to the two craft practitioners. A comprehensive understanding of the knowledge systems that underlie cultural crafts, allows us to most efficiently and effectively work with them as those knowledge systems inform the formal models of our software.

Research Question 2: How do we transform the formal models gained in Question 1 into user friendly, pedagogical software?

Once we arrived at the formal models (i.e. outcome from question 1), we needed to translate them into a user-friendly, software interface that served a pedagogical purpose. In our case, pCSDTs that 1) students would enjoy using, 2) provided a faithful as possible recreation of the indigenous craft or cultural practice in simulation, and 3) served a pedagogical purpose. This 'sweat spot', as we have come to call it, is sometimes very challenging to find, as we saw with both the Kente Cloth and Adinkra stamping

pCSDTs. Of the two, I think that I am the most proud of the Adinkra stamping pCSDT. The Adinkra applet succeeded at finding the 'sweat spot' between a well-functioning user interface, the complex mathematics of logarithmic spirals, and the scripting requirements found in computer science education. Achieving the 'sweet spot' where student interest, cultural authenticity, and curricular demands successfully intersected for the Adinkra pCSDT was one of the most rewarding challenges for me.

Research question 3: How do we avoid the ecosystem colonization of our software that other pedagogical software have undergone?

Deciding the proper balance for 'locking' down the design tools was surprisingly difficult. Determining the amount of extensibility allowed to students, in adding functionality that is unrelated to the cultural crafts and practices under simulation, needed just the right balance. Restricting this ability with a Tool to too great of an extent, would make it seem boring, however, no restrictions would run the risk of corporate or violent colonization as explored in chapter 4.

I think the Cornrows applet is a good example of such a balance that we successfully negotiated. In Cornrows, the user is able to upload their own graphics files to make braids of their friend's faces or other images, but the braiding infrastructure does not change. Thus, this is an example where the tool is extensible in creative ways; however, it remains a tool for cultural simulation.

Research question 4: How can we use formative feedback to inform our development efforts?

As we have seen in chapter 5 of this work, user and developer ethnographies have been crucial in informing our development decisions on the pCSDTs. Using formative feedback, the interface design gradually evolved in an 'Agile' model of iterative development. Our development meetings each week included a review of the progress made during the previous week. Following this review, we made decisions on what next feature or 'bug' should receive attention, in light of student usability testing that was simultaneously being conducted.

Research question 5: How do we conduct a summative evaluation of our software with student users, as a final check on the viability of our software?

Summative evaluation was used as a final check on the viability of the software in meeting our goals of improving the math and computing performance and interest of underrepresented students, as well as for students in a 'postcolonial' site in West Africa. As recounted in chapter 6, we conducted an experiment at a junior high school in Kumasi Ghana using the current version of the pCSDTs, called CSnap. We have named the lesson that we tested 'Adinkra Mathematics', where we taught the mathematics of logarithmic spirals using both a non-culture based computer program and our CSnap application. We view the statistical outcome of that randomized experiment as highly significant and are greatly encouraged to move forward with further inquiry in this area.

7.2 Contributions to Society

When we consider the successful outcomes that we have achieved through this research, we need to remind ourselves of our initial motivations. The U.S. is plagued by historical harms that began with European imperialism and colonization and which present today as social exclusions and systemic inequalities, such as what is described as the 'quiet crisis' of underrepresentation in the STEM fields. Ethnocomputing can be used a way of creating 'content aware' 'user spaces' in which indigenous knowledge systems and culturally relevant knowledge systems become a sort of cultural capital as a way to counter prevalent internalized pathologies of the myths of genetic and cultural determinism that give rise to STEM avoidance. Culture inclusive educational tools, such as the pCSDTs represent a constructionist pedagogy that has been uniquely designed to fill potentially harmful spaces with culturally significant computational capital. The content-aware approach used in the pCSDTs has the advantage of a built-in anti-racist message, with the potential to decolonize computing and mathematics education because of the usefulness of its ability to both be inclusive and bridging. Because of this unique approach, content-aware software, such as the pCSDTs, can also be a way to address and heal historical harms. My motivation for doing this research is to help improve educational outcomes for underrepresented students.

It is my opinion that our research shows that ethnocomputing can play a very effective role in improving student outcomes. We (and others) have shown that computational thinking results when students interact deeply with constructionist base

software such as the pCSDTs. We have a strong experimental result that shows that test scores do improve with these culture-based simulations. I hope to have the opportunity to continue working on these types of projects in the future.

References

[1] S. Jackson, "Waking up to the quiet crisis in the United States: Its time for a new call to action.," *The College Board Review*, vol. 210, pp. 24-27, Winter/Spring 2007.

[2] Nat. Sci. Found. (2011, Jan. 12). *Women, Minorities, and Persons with Disabilities in Science and Engineering: 2011* [Online]. Available: http://www.nsf.gov/statistics/wmpd/, Date Last Accessed 11/02/2014.

[3] J. Graves. (2010, Oct. 20). *How does creationism harm African Americans?* [Online]. Available: http://evostudies.org/2010/06/how-does-creationism-harm-african-americans/, Date Last Accessed 11/02/2014.

[4] I. Olalde, M. E. Allentoft, F. Sánchez-Quinto, G. Santpere, C. W. Chiang, M. DeGiorgio, *et al.*, "Derived immune and ancestral pigmentation alleles in a 7,000-year-old Mesolithic European," *Nature*, vol. 507, no. 7491, pp. 225-228, Jan. 2014.

[5] J. Tooby and L. Cosmides, "The psychological foundations of culture," in *The Adapted Mind: Evolutionary Psychology and the Generation of Culture*, J. H. Barkow, L. Cosmides, and J. Tooby, Eds., New York, NY, USA: Oxford Univ. Press, 1995, pp. 19-136.

[6] S. J. Gould, *The Mismeasure of Man*. New York, NY, USA: Norton, 1996.

[7] J. U. Ogbu and H. D. Simons, "Voluntary and involuntary minorities: A cultural-ecological theory of school performance with some implications for education," *Anthropology & Educ. Quart.*, vol. 29, no. 2, pp. 155-188, June 1998.

[8] S. Fordham, "Peer-proofing academic competition among black adolescents: "Acting white" black American style," in *Empowerment Through Multicultural Education*, C. E. Sleeter, Ed., Albany, NY, USA: State Univ. of NY Press, 1991, pp. 69-93.

[9] R. G. Fryer Jr and P. Torelli, "An empirical analysis of acting white," *J. of Public Econ.*, vol. 94, no. 5, pp. 380-396, June 2010.

[10] C. M. Steele, S. J. Spencer, and J. Aronson, "Contending with group image: The psychology of stereotype and social identity threat," *Advances in Experimental Social Psychology*, vol. 34, pp. 379-440, 2002.

[11] M. L. Andersen and P. H. Collins, *Race, Class, and Gender: An Anthology*, 6th ed. Belmont, CA, USA: Wadsworth, 2007.

[12] R. A. Olson. (2014, Oct. 20). *White privilege in schools* [Online]. Available: http://www.createwisconsin.net/events/ConferenceHandouts/Tuesday/845am/White_Privilege_in_Schools.pdf, Date Last Accessed 11/02/2014.

[13] J. R. Flynn, *Are We Getting Smarter? Rising IQ in the Twenty-first Century*. New York, NY, USA: Cambridge Univ. Press, 2012.

[14] M. Gladwell. (2007, Dec. 17). None of the above: What IQ doesn't tell you about race. *The New Yorker* [Online]. 31-34. Available: http://www.newyorker.com/magazine/2007/12/17/none-of-the-above, Date Last Accessed 11/02/2014.

[15] E. Roivainen, "Economic, educational, and IQ gains in eastern Germany 1990–2006," *Intelligence*, vol. 40, no. 6, pp. 571-575, Dec. 2012.

[16] P. Bourdieu, "The forms of capital," in *Readings in Economic Sociology*, N. W. Biggart, Ed., Malden, MA, USA: Blackwell, 2002, pp. 280-291.

[17] K. Brennan and M. Resnick, "New frameworks for studying and assessing the development of computational thinking," in *Proc. of the 2012 Annu. Meeting of the Amer. Educational Research Assoc.*, Vancouver, Canada, 2012, pp. 1-25.

[18] A. Pickering, *The Mangle of Practice: Time, Agency, and Science*. Chicago, IL, USA: Univ. of Chicago Press, 1995.

[19] A. Bennett and R. Eglash, "cSELF (Computer science education from life): Broadening participation through design agency," *Int. J. of Web-Based Learning and Teaching Technologies (IJWLTT)*, vol. 8, no. 4, pp. 34-49, Oct.-Dec. 2013.

[20] W. M. Geniusz, *Our Knowledge is not Primitive: Decolonizing Botanical Anishinaabe Teachings*, 1st ed. Syracuse, NY, USA: Syracuse Univ. Press, 2009.

[21] B. Babbitt, D. Lyles, and R. Eglash, "From ethnomathematics to ethnocomputing," in *Alternative Forms of Knowing (in) Mathematics*, S. Mukhopadhyay and W.-M. Roth, Eds., Rotterdam, The Netherlands: Sense Publishers, 2012, pp. 205-219.

[22] W. Babbitt, M. Lachney, E. Bulley, and R. Eglash, "Adinkra mathematics: A randomized, controlled study of ethnocomputing in Ghana," *For the Learning of Math.*, submitted for publication.

[23] J. M. Jama, "The role of ethnomathematics in mathematics education cases from the horn of Africa," *ZDM*, vol. 31, no. 3, pp. 92-95, June 1999.

[24] P. Gerdes, "Conditions and strategies for emancipatory mathematics education in undeveloped countries," *For the Learning of Math.*, vol. 5, no. 1, pp. 15-20, Feb. 1985.

[25] S. Anderson, "Worldmath curriculum: Fighting eurocentrism in mathematics," *J. of Negro Educ.*, vol. 59, no. 3, pp. 348-359, Summer 1990.

[26] M. Frankenstein and A. B. Powell, "Toward liberatory mathematics: Paulo Freire's epistemology and ethnomathematics," in *Politics of Liberation: Paths From Freire*, C. Lankshear and P. McLaren, Eds., London, UK: Routledge, 1994, pp. 74-99.

[27] C. Zaslavsky, "'Africa Counts' and ethnomathematics," *For the Learning of Math.*, vol. 14, no. 2, pp. 3-8, June 1994.

[28] S. G. Harding, *Sciences From Below: Feminisms, Postcolonialities, and Modernities*. Durham, NC, USA: Duke Univ. Press, 2008.

[29] F. Elliott, "Science, metaphoric meaning, and Indigenous Knowledge," *Alberta J. of Educational Research*, vol. 55, no. 3, pp. 284-297, Fall 2009.

[30] F. Ahia and E. Fredua-Kwarteng, "Gazing mathematics and science education in Ghana," in *Contemporary Issues in African Sciences and Science Education*, A. Asabere-Ameyaw, G. J. Sefa Dei, K. Raheem, and J. Anamuah-Mensah, Eds., Rotterdam, The Netherlands: Sense Publishers, 2012, pp. 103-125.

[31] A. N. Ezeife, "Using the environment in mathematics and science teaching: An African and aboriginal perspective," *Int. Review of Educ.*, vol. 49, no. 3-4, pp. 319-342, July 2003.

[32] J. Lipka, M. P. Hogan, J. P. Webster, E. Yanez, B. Adams, S. Clark, et al., "Math in a cultural context: Two case studies of a successful culturally based math project," *Anthropology & Educ. Quart.*, vol. 36, no. 4, pp. 367-385, Jan. 2005.

[33] R. Eglash, M. Krishnamoorthy, J. Sanchez, and A. Woodbridge, "Fractal simulations of African design in pre-college computing education," *ACM Trans. on Computing Educ. (TOCE)*, vol. 11, no. 3, article 17, Oct. 2011.

[34] S. Khan, "Ethnomathematics as mythopoetic curriculum," *For the Learning of Math.*, vol. 31, no. 3, pp. 14-18, Nov. 2011.

[35] U. D'Ambrosio, "From Ea, through Pythagoras, to Avatar: Different settings for mathematics," in *Proc. of the 34th Conf. of the Int. Group for the Psychology of Math. Educ.*, 2010, pp. 1-20.

[36] R. Eglash and C. Garvey, "Basins of attraction for generative justice," in *Chaos Theory in Politics*, S. Banerjee, S. S. Ercetin, and A. Tekin, Eds., The Netherlands: Springer, 2014, pp. 75-88.

[37] W.-M. Roth and A. C. Barton, *Rethinking Scientific Literacy*, 1st ed. New York, NY, USA: Routledge, 2004.

[38] E. Gutstein, *Reading and Writing the World with Mathematics: Toward a Pedagogy for Social Justice.* New York, NY, USA: Routledge, 2006.

[39] U. D'Ambrosio, "Ethnomathematics: A response to the changing role of mathematics in society," *Philosophy of Math. Educ.*, vol. 25, Oct. 2010.

[40] M. Rosa and D. Clark, "Ethnomodeling as a pedagogical tool for the ethnomathematics program," *Revista Latinoamericana de Etnomatemática*, vol. 3, no. 2, pp. 14-23, 2010.

[41] J. M. Wing, "Computational thinking," *J. of Computing Sciences in Colleges*, vol. 24, no. 6, pp. 6-7, June 2006.

[42] J. Margolis, R. Estrella, J. Goode, J. J. Holme, and K. Nao, *Stuck in the Shallow End*. Cambridge, MA, USA: MIT Press, 2008.

[43] R. Eglash, "Native-American analogues to the Cartesian coordinate system," in *Culturally Responsive Mathematics Education*, B. Greer, S. Mukhopadhyay, A. B. Powell, and S. Nelson-Barber, Eds., New York, NY, USA: Routledge, 2009, pp. 281-294.

[44] J. Barta and R. Eglash, "Teaching artful expressions of mathematical beauty: Virtually creating Native American beadwork and rug weaving," in *Handbook of Research on Computational Arts and Creative Informatics.*, J. Braman, Ed., Hershey, PA, USA: IGI Global, 2009, pp. 280-289.

[45] R. Eglash, *African Fractals: Modern Computing and Indigenous Design.* Piscataway, NJ, USA: Rutgers Univ. Press, 1999.

[46] *Inquiring into Inquiry: Learning and Teaching in Science.* Washington, DC: AAAS, 2000.

[47] T. A. Brush and J. W. Saye, "A summary of research exploring hard and soft scaffolding for teachers and students using a multimedia supported learning environment," *The J. of Interactive Online Learning*, vol. 1, no. 2, pp. 1-12, Fall 2002.

[48] M. Resnick, J. Maloney, A. Monroy-Hernández, N. Rusk, E. Eastmond, K. Brennan, et al., "Scratch: Programming for all," *Commun. of the ACM*, vol. 52, no. 11, pp. 60-67, Nov. 2009.

[49] R. S. Rattray, G. T. Bennett, V. Blake, H. D. Buxton, R. R. Marett, and C. G. Seligman, *Religion and Art in Ashanti*. Oxford, UK: Clarendon, 1927.

[50] *Kente Cloth Patterns and Their Meanings* [Online]. 2012. Available: http://csdt.rpi.edu/african/kente/patterns.html, Date Last Accessed 11/02/2014.

[51] W. B. Willis, *The Adinkra Dictionary: A Visual Primer on the Language of Adinkra*. Washington, DC, USA: Pyramid Complex, 1998.

[52] T. A. Cook, *The Curves of Life: Being an Account of Spiral Formations and Their Application to Growth in Nature, to Science, and to Art: With Special Reference to the Manuscripts of Leonardo da Vinci.* New York, NY, USA: Dover, 1979.

[53] W. Babbitt, "An analysis of the programmable Culturally Situated Design Tools from an HCI perspective," presented at the *3rd Annu. Symp., Theory and Research in HCI*, Troy, NY, USA, 2012.

[54] R. Eglash and A. Bennett, "Teaching with hidden capital: Agency in children's computational explorations of cornrow hairstyles," *Environments*, vol. 19, no. 1, pp. 58-73, 2009.

[55] L. S. Vygotsky, *Mind in Society: The Development of Higher Psychological Processes*, 14th ed. Cambridge, MA, USA: Harvard Univ. Press, 1978.

[56] M. Resnick and B. Silverman, "Some reflections on designing construction kits for kids," in *Proc. of the 2005 Conf. on Interaction Design and Children*, Boulder, CO, USA, 2005, pp. 117-122.

[57] M. Fowler and J. Highsmith, "The Agile Manifesto," *Software Develop.*, vol. 9, no. 8, pp. 28-35, Aug. 2001.

[58] Student, private communication, 2012.

[59] P. N. Johnson-Laird, "Mental models and deductive reasoning," in *Reasoning: Studies in Human Inference and its Foundations.* , J. E. Adler and L. J. Rips, Eds., Cambridge, UK: Cambridge Univ. Press, 2008, pp. 206-222.

[60] P. Turner and G. Walle, "Familiarity as a basis for universal design," *Gerontechnology*, vol. 5, no. 3, pp. 150-159, Sept. 2006.

[61] D. Rapp, "Mental models: Theoretical issues for visualizations in science education," in *Visualization in Science Education*, J. K. Gilbert, Ed., Dordrecht, The Netherlands: Springer, 2005, pp. 43-60.

[62] K. Craik, *The Nature of Exploration.* Cambridge, UK: Cambridge Univ. Press, 1943.

[63] Apple Inc., *Macintosh Human Interface Guidelines.* New York, NY, USA: Addison-Wesley, 1992.

[64] M. Lachney, W. Babbitt, and R. Eglash, "Alternatives to the content agnostic position in the 'Construction Genre' of learning technology," *Computational Culture: A J. of Software Stud.*, submitted for publication.

[65] M. Ito, *Engineering Play: A Cultural History of Children's Software*. Cambridge, MA, USA: The MIT Press, 2009.

[66] I. Harel and S. Papert, Eds., *Constructionism*. Westport, CT, USA: Ablex, 1991.

[67] S. Papert, "Perestroika and epistemological politics," in *Constructionism*, S. Papert and I. Harel, Eds., Westport, CT, USA: Ablex, 1991, pp. 13-28.

[68] C. K. Olson, L. A. Kutner, D. E. Warner, J. B. Almerigi, L. Baer, A. M. Nicholi II, et al., "Factors correlated with violent video game use by adolescent boys and girls," *J. of Adolescent Health*, vol. 41, no. 1, pp. 77-83, July 2007.

[69] T. Bruce, *Time to Play in Early Childhood Education*. London, UK: Hodder & Stoughton, 1991.

[70] S. Smilansky and L. Shefatya, Facilitating Play: A Medium for Promoting *Cognitive, Socio-Emotional, and Academic Development in Young Children*. Gaithersburg, MD, USA: Psychosocial & Educational Publications, 1990.

[71] K. Sanford and L. Madill, "Resistance through video game play: It's a boy thing," *Canadian J. of Educ./Revue Canadienne de L'éducation*, vol. 29, no. 1, pp. 287-306, 2006.

[72] B. Beal, "Alternative masculinity and its effect on gender relations in the subculture of skateboarding," *J. of Sport Behavior*, vol. 19, no. 3, pp. 204-220, Aug. 1996.

[73] C. Bacon-Smith, *Enterprising Women: Television Fandom and the Creation of Popular Myth*. Philadelphia, PA, USA: Univ. of Pennsylvania Press, 1992.

[74] H. Jenkins, *Textual Poachers: Television Fans & Participatory Culture*. New York, NY, USA: Routledge, 2013.

[75] C. Penley, *NASA/Trek: Popular Science and Sex in America*. New York, NY, USA: Verso, 1997.

[76] S. Ferguson, "The children's culture industry and globalization: Shifts in the commodity character of toys," in *Int. Symp. "Transformations in the Cultural and Media Industries,"* 2006, pp. 1-11.

[77] R. L. Hasen, "Citizens United and the Orphaned Antidistortion Rationale," *Ga. St. UL Rev*., vol. 27, p. 989, 2010.

[78] S. Papert, *Mindstorms: Children, Computers, and Powerful Ideas*. New York, NY, USA: Basic Books, 1980.

[79] H. Abelson and A. Di Sessa, *Turtle Geometry: The Computer as a Medium for Exploring Mathematics.* Cambridge, MA, USA: MIT Press, 1986.

[80] U. Wilensky, "Making sense of probability through paradox and programming," in *Constructionism in Practice: Designing, Thinking, and Learning in a Digital World*, Y. B. Kafai and M. Resnick, Eds., Mahwah, NJ, USA: Erlbaum, 1996, pp. 269-296.

[81] M. Gura, *Getting Started with LEGO Robotics: A Guide for K-12 Educators*. Eugene, OR, USA: International Society for Technology in Education, 2011.

[82] J. Piaget, *The Construction of Reality in The Child*. Oxford, UK: Routledge, 1954.

[83] S. A. Rose and M. Blank, "The potency of context in children's cognition: An illustration through conservation," *Child Develop.*, vol. 45, no. 2, pp. 499-502, June 1974.

[84] B. Inhelder and J. Piaget, *The Growth of Logical Thinking from Childhood to Adolescence.* Oxford, UK: Routledge, 1958.

[85] S. Turkle and S. Papert, "Epistemological pluralism and the revaluation of the concrete," in *Constructionism*, I. Harel and S. Papert, Eds., Westport, CT, USA: Ablex, 1991, pp. 161-191.

[86] E. F. Keller, *A Feeling for the Organism: The Life and Work of Barbara McClintock.* New York, NY, USA: Henry Holt, 1983.

[87] C. Gilligan, *In a Different Voice: Psychological Theory and Women's Development*. Cambridge, MA, USA: Harvard Univ. Press, 1982.

[88] S. Papert, "Teaching children thinking," *Contemporary Issues in Technology and Teacher Educ.*, vol. 5, no. 3, pp. 353-365, July 2005.

[89] R. Eglash and D. Banks, "Recursive depth in generative spaces: Democratization in three dimensions of technosocial self-organization," *The Inform. Soc.*, vol. 30, no. 2, pp. 106-115, Mar. 2014.

[90] B. Latour, *We Have Never Been Modern*. Cambridge, MA, USA: Harvard Univ. Press, 1993.

[91] C. Bowers and F. Apffel-Marglin, Eds., *Rethinking Freire: Globalization and the environmental crisis*. Mahwah, NJ, USA: Erlbaum, 2005.

[92] M. Foucault, *The Archaeology of Knowledge*. New York, NY, USA: Harper & Row, 1972.

[93] S. Traweek, *Beamtimes and Lifetimes: The World of High Energy Physicists.* Cambridge, MA, USA: Harvard Univ. Press, 1988.

[94] NGSS. (2013, Jan. 21). *Next Generation Science Standards* [Online]. Available: http://www.nextgenscience.org/next-generation-science-standards, Date Last Accessed 11/02/2014.

[95] N. Gonzalez, L. C. Moll, and C. Amanti, Eds., *Funds of Knowledge: Theorizing Practices in Households, Communities, and Classrooms*. Mahwah, NJ, USA: Erlbaum, 2005.

[96] NABT. (2008, Jan. 11). *Sustainability in life science teaching* [Online]. Available: http://www.nabt.org/websites/institution/index.php?p=520, Date Last Accessed 11/02/2014.

[97] NCTM. (2013, Jan. 11). *Connections* [Online]. Available: http://www.nctm.org/standards/content.aspx?id=26855, Date Last Accessed 11/02/2014.

[98] CSTA. (2012, Sept. 11). *CSTA K-12 Computer Science Standards, Rev. 2011* [Online]. 2014. Available: http://csta.acm.org/Curriculum/sub/CurrFiles/CSTA_K-12_CSS.pdf, Date Last Accessed 11/02/2014.

[99] M. Ascher, *Ethnomathematics: A Multicultural View of Mathematical Ideas*. Boca Raton, FL, USA: CRC Press, 1994.

[100] R. Lara-Alecio, B. J. Irby, and L. Morales-Aldana, "A mathematics lesson from the Mayan civilization," *Teaching Children Math.*, vol. 5, no. 3, pp. 154-159, Nov. 1998.

[101] M. M. Jennings. “Rain-forest algebra and MTV geometry,” in *The Textbook Letter* [Online]. Nov.-Dec., 1996. Available: http://www.textbookleague.org/75math.htm, Date Last Accessed 11/02/2014.

[102] B. Zolkower, "Math fictions," in *Technoscience and Cyberculture*, S. Aronwitz, B. Martinsons, and M. Menser, Eds., New York, NY, USA: Routledge, 1996, pp. 57-96.

[103] R. Eglash, "When math worlds collide: Intention and invention in ethnomathematics," *Sci., Technology & Human Values*, vol. 22, no. 1, pp. 79-97, Winter 1997.

[104] R. Eglash, "Race, sex, and nerds: From black geeks to Asian American hipsters," *Social Text*, vol. 20, pp. 49-64, Summer 2002.

[105] J. Lipka, J. Webster, and E. Yanez, "Factors that affect Alaska Native students' mathematical performance [Special issue]," *J. of Amer. Indian Educ.*, vol. 44, no. 3, pp. 1-8, 2005.

[106] E. Harrison. (2011, Nov. 24). *From Ordinary Shopper To Celebrity, Overnight* [Online]. Available: http://www.npr.org/2011/01/05/132379365/from-ordinary-shopper-to-celebrity-overnight, Date Last Accessed 11/02/2014.

[107] D. Oyserman, K. Harrison, and D. Bybee, "Can racial identity be promotive of academic efficacy?," *Int. J. of Behavioral Develop.*, vol. 25, no. 4, pp. 379-385, Aug. 2001.

[108] I. Altschul, D. Oyserman, and D. Bybee, "Racial-Ethnic identity in mid-adolescence: Content and change as predictors of academic achievement," *Child Develop.*, vol. 77, no. 5, pp. 1155-1169, June 2006.

[109] A. Celious and D. Oyserman, "Race from the inside: An emerging heterogeneous race model," *J. of Social Issues*, vol. 57, no. 1, pp. 149-165, Spring 2001.

[110] R. Eglash, J. E. Gilbert, V. Taylor, and S. R. Geier, "Culturally responsive computing in urban, after-school contexts two approaches," *Urban Educ.*, vol. 48, no. 5, pp. 629-656, Sept. 2013.

[111] K. Eyferth, "Leistungen verschidener Gruppen von Besatzungskindern in Hamburg-Wechsler Intelligenztest fur Kinder (HAWIK) [Performance of different groups of occupation children on the Hamburg-Wechsler Intelligence test for children]," *Archhiv fur die gesamte Psychologie*, no. 113, pp. 222-241, 1961.

[112] P. Arcidiacono, A. Beauchamp, M. Hull, and S. Sanders. (2011, Jan. 11). *Isolating Mechanisms for the Racial Divide in Education and the Labor Market:*

Evidence From Interracial Families [Online]. Available: http://www.aeaweb.org/aea/2014conference/program/retrieve.php?pdfid=836, Date Last Accessed 01/11/2014.

[113] P. Freire, *Pedagogy of the Oppressed*. New York, NY, USA: Bloomsbury, 2000.

[114] E. Gutstein, "Teaching and learning mathematics for social justice in an urban, Latino school," *J. for Research in Math. Educ.*, vol. 34, no. 1, pp. 37-73, Jan. 2003.

[115] M. Adams, "Hybridizing habitus and reflexivity: Towards an understanding of contemporary identity?," *Sociology*, vol. 40, no. 3, pp. 511-528, June 2006.

[116] J. J. Ryoo, J. Margolis, C. H. Lee, C. D. Sandoval, and J. Goode, "Democratizing computer science knowledge: Transforming the face of computer science through public high school education," *Learning, Media and Technology*, vol. 38, no. 2, pp. 161-181, Jan. 2013.

[117] K. A. Scott and M. A. White, "COMPUGIRLS' standpoint culturally responsive computing and its effect on girls of color," *Urban Educ.*, vol. 48, no. 5, pp. 657-681, Sept. 2013.

[118] M. Tedre and R. Eglash, "Ethnocomputing," in *Software Studies: A Lexicon*, M. Fuller, Ed., Cambridge, MA, USA: The MIT Press, 2008, pp. 92-101.

[119] M. A. Millerick, "Multicultural students' perspectives on their mathematics education," M.S. thesis, School of Educ., Dominican Univ. of California San Rafael, CA, 2008.

[120] R. Eglash, A. Bennett, C. O Donnell, S. Jennings, and M. Cintorino, "Culturally Situated Design Tools: Ethnocomputing from field site to classroom," *Amer. Anthropologist*, vol. 108, no. 2, p. 347, June 2006.

[121] P. E. Willis, *Learning to Labor: How Working Class Kids Get Working Class Jobs*. New York, NY, USA: Columbia Univ. Press, 1977.

[122] P. Bourdieu and L. J. D. Wacquant, *An Invitation to Reflexive Sociology*. Chicago, IL, USA: Univ. of Chicago Press, 1992.

[123] K. D. Gutiérrez and B. Rogoff, "Cultural ways of learning: Individual traits or repertoires of practice," *Educational Researcher*, vol. 32, no. 5, pp. 19-25, June 2003.

[124] M. Pollock, "Race bending: Mixed youth practicing strategic racialization in California," *Anthropology & Educ. Quart.*, vol. 35, no. 1, pp. 30-52, Jan. 2004.

[125] M. Pollock, *Everday Antiracism: Getting Real About Race in School*. New York, NY, USA: New Press 2008.

[126] J. Blomberg and M. Burrell, "An ethnographic approach to design," in *Human-Computer Interaction: Development Process*, A. Sears and J.A. Jacko, Eds., Boca Raton, FL, USA: CRC Press, 2009, pp. 71-94.

[127] G. B. West, J. H. Brown, and B. J. Enquist, "A general model for the origin of allometric scaling laws in biology," *Sci.*, vol. 276, no. 5309, pp. 122-126, Apr. 1997.

[128] T. D. Cook, D. T. Campbell, and A. Day, *Quasi-Experimentation: Design & Analysis Issues for Field Settings.* Boston, MA, USA: Houghton Mifflin, 1979.

Appendix 1 Culture vs. Non-Culture Pre/Post Test

1) Define the following terms in your own words. You may draw images to help you explain.

a) Exponential Growth

b) Exponential Parameter

c) Tangent Vector

d) Angle

e) Degree

f) Cartesian Plane

2) Describe how you would create the following figure using math or computing.

3) Name three examples of logarithmic spirals in nature.

a)

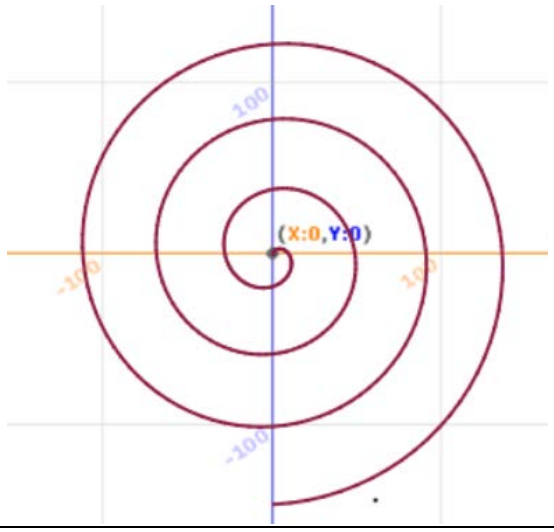
b)

c)

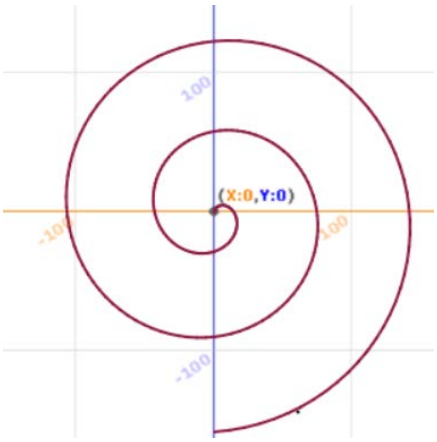
4) Look at spirals a, b, and c below. Put the spirals in order from the smallest to the largest exponential parameters. 1 being the smallest and 3 being the largest.

1. _____ 2. _____ 3. _____

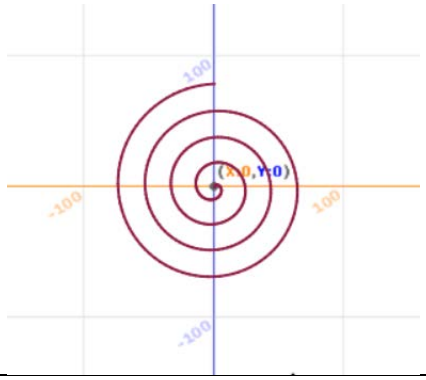
a.



b.



c.



5) Describe the shape below mathematically.

6) Give three examples of logarithmic spirals in everyday life.

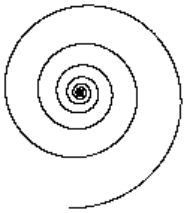
a)

b)

c)

7) Look at shapes a and b, which shape is found more in nature? Circle the answer below.

a. 

b. 

c. Neither shape is found in nature.

8) Draw two circles that intersect from the tangent vectors below.