Craft of Computing: Using a Novel Domain to Broaden Undergraduate Participation and Perceptions of Computing at the CS0 Level

Emily Lovell
Computer Science Department
UC Santa Cruz / Berea College
Santa Cruz, CA / Berea, KY
emme@soe.ucsc.edu

James Davis
Computer Science Department
UC Santa Cruz
Santa Cruz, CA
davis@soe.ucsc.edu

***Abstract*—In this Innovative Practice Full Paper, we report on a CS0-level computational craft course added to our departmental offerings in hopes of further broadening participation. We summarize the course design and structure, which emphasize algorithmic design (using Processing), handcraft, and digital fabrication. We share examples of creative computational work and feedback from students, as well as reflections on the course's efficacy within our funnel-style curriculum. Early evidence suggests that the course offers a highly personal and creative entry point to computing – and one that is effective at engaging a diversity of students while ensuring a smooth transition to CS1.**

***Keywords*—Computer science, higher education, student diversity, underrepresentation, course design, instructional design, creativity, recruitment and retention**

I. Introduction

CS0 courses can offer students with little-to-no computing background the opportunity to explore computer science before committing to a major [1]. What's more, CS0 courses can help to level the playing field for these students by the time they enter CS1 alongside peers who may have taken computer science courses in high school and/or have a stronger mathematics background [2].

Berea College, among other institutions, has adopted a CS0 "funnel" approach to attract minoritized students; that is to say, we offer several CS0 courses on topics of interest to the general student population [3]. This approach has since been adopted and found to be effective elsewhere [1], [4]. Upon her hire, one of the authors was invited to design and teach a new CS0 course in her domain of expertise – computational craft – in hopes of further broadening departmental demographics. Computational craft has been studied as a successful avenue for attracting and retaining groups historically excluded from computing, particularly women [5], [6]. This course, entitled *Craft of Computing*, covers core CS0 concepts including computational thinking, variables, loops, functions, etc. The course also showcases the creative possibilities of computing when paired with handcraft, digital fabrication, and algorithmic design. *Craft of Computing* was taught five times over a two-year period, during which it was deemed so successful as to be added to the college's permanent catalog.

In this paper, we detail the course structure, learning goals, and major assignments – complete with many compelling examples of student work; *Craft of Computing* students have created personally relevant and meaningful artifacts, often displayed in their dorm rooms or given as gifts. Examples include: a needle-felted fairy house with embedded LED lights, light-up paper circuit valentines, and beautiful recursive geometric patterns – first generated in Processing and then realized in the form of plotter drawings, vinyl-cut laptop stickers, and laser-cut wooden coasters.

Early analysis suggests that the course offers a highly personal and creative entry point to computing – and one that is effective at engaging a diversity of students while ensuring a smooth transition to CS1. Our personal observations are supplemented with student feedback in the form of interviews, informal course reflections, and end-of-term course evaluations. We also provide insights and recommendations for others looking to adopt a craft-themed CS0 course.

In sum, the primary goal of this paper is to document *Craft of Computing*, a novel undergraduate course designed to broaden *perceptions* about computing – which in turn, influences who *participates* in computing [7], [8].

II. Related Work

Our work builds upon a body of prior research on broadening participation in computing, CS0 course design and outcomes, the Processing programming language, and computational craft.

The dearth of women and other minoritized groups in computing is well documented, as are some of the factors critical to attracting and retaining them [9], [10]. Most relevant to this paper are a sense of belonging and engaging in work that is personally meaningful and/or culturally relevant [11], [12], [13], [14]. Building technological self-efficacy and a growth mindset can further support these goals [15], [16], [17], [18]. The design of *Craft of Computing* builds upon this knowledge, including the incorporation of specific recommendations – for example, encouraging students to pursue identity-affirming

creative work and curating a physical space in which students of diverse identities feel welcome [19].

There is also ample research on the importance of CS0-level courses as an avenue into computing, especially for students with little-to-no prior programming experience. Prior work also affirms the importance of CS0 as a way to broaden student perceptions of computer science [1]. To this end, a "funnel" curricular model has been well-documented, in which several different themed CS0 courses are offered as a means to invite diverse participation [1], [3], [4]. Our work expands existing practice by offering yet another novel entry point to computing, through a domain which is both creative in nature and stereotypically "softer": craft.

The Processing programming language was designed to support exactly this kind of creative work [20], [21], [22]; it was developed by designers and meant to be more accessible than it's closest counterpart, Java. Processing has since been leveraged in university-level offerings of "CS Principles" and CS0-style courses, both as an approachable text-based language and a means to create art [23], [24]. It is for exactly these reasons that *Craft of Computing* is taught using Processing.

Computational craft – especially the field of electronic textiles – has been established as an effective means to broaden participation in computing, especially at the K-12 level and in after-school settings [6], [25], [26]. Our work expands these initiatives to the undergraduate context, in which students are critically deciding upon and pursuing a field of study to propel their careers. Our work also infuses a stronger software component (via programming in Processing), in hopes that this may prepare students for CS1.

Lastly, the design of this course was especially informed by the work of CU Boulder's Craft Technology Lab and the MIT Media Lab's High-Low Tech research group [27]. Both of these now-defunct groups have laid a strong foundation of tutorials and tools which enable making, hacking, and programming rooted in craft. Their missions focused on democratization of engineering and the radical inclusion of diverse populations – and their research and teaching has very directly inspired our efforts to formalize a course at the *undergraduate* level.

III. Background

Berea College is a small liberal arts institution, which is also a work college. The college's mission is to serve students of great academic promise, but limited economic means; all students attend on full scholarship and participate in a campus labor program, through which they work part-time for the college while pursuing their degrees. It is a unique and impactful setting in which to consider broadening participation in computing, as our student body is extremely diverse and largely hails from historically underserved communities.

Our course joined an existing selection of themed CS0 offerings at the college, all of which funnel into a singular CS1 course. Examples include: *Intro to Robotics, Storytelling with Alice, Intro to Game Design*, and *Building Better Apps*. While *Craft of Computing* debuted as a "special topics"/elective offering of this variety, it was added to the college's permanent catalog within one year.

Craft of Computing was taught five times over a period of four consecutive 15-week academic terms. This paper includes examples of student work across all of the terms in which the course was offered. However, in detailing the structure of this course, this paper will focus on the latest iteration unless otherwise noted. There were some notable changes made over the two year period in which the course was developed; those are discussed toward the end of this paper in the context of recommendations for others.

IV. Course Design

Craft of Computing shares the same learning goals of many other CS0-level courses, namely to teach core computer science competencies while showcasing applications of computing (in this case, creative ones), and to lower the barriers to entry for students with little or no programming experience [1]. We leverage craft as a context because it is both relatable and provocative when considered in juxtaposition to computing – while also exposing students to creative applications of programming. In terms of domain-specific content, *Craft of Computing* exposes students to the following:

- *Programming in Processing* – including coverage of computational concepts such as loops, variables, and functions (facilitating the creation of computational art)
- *Some basic electronics* – including simple textile and paper circuits
- *Handcraft* – including needle felting and embroidery (allowing students to realize their algorithmic designs in a traditional craft medium)
- *Digital fabrication* – including use of a plotter, laser cutter, and vinyl cutter (allowing students to realize their algorithmic designs in a computer-mediated craft medium)

The course also includes some coverage of the *maker movement*, in particular, discussion around accessibility and inclusivity of maker culture.

A. Setting & Organization

Craft of Computing is taught in a lab space shared by our electronics course, which allows access to all of the relevant tools, materials, and equipment, as well as a sink for cleanup and a safe place for students to leave projects-in-progress. This classroom is also the setting for our department's evening lab hours which are open to students in all computer science courses, and where students may drop in for help or to work on projects. An assortment of TAs staff this space Sunday through Thursday night each week, and an effort is made to have one or more TAs from each class scheduled on any given night. Students feel great ownership over this space.

The course meets for long class periods – 110 minutes – twice per week. One day per week focuses on programming or computational concepts, such as Processing syntax, coordinate systems, and programming fundamentals. These class periods

Assignments	20%	Includes written reading responses, programming exercises, and outside-of-class research. Intended to broaden both craft and technical skill sets.
Quizzes (x5)	25%	Cover technical content and allow the instructor to better understand which topics would benefit from more coverage. Quizzes are reviewed in class, with focus on revisiting topics students may have struggled with.
Mini-projects (x3)	30%	Through these special assignments, students learn to interface between the digital and physical worlds — for example, by making something on a vinyl cutter which was first designed on a computer using Processing.
Final project	25%	Invites students to blend craft and computation in a more open-ended context than the mini-projects. Once again, students bring a design of their own into the physical world — but they do so through a medium and technique of their choice.

Fig. 1. Coursework components, with percentage of overall course grade and a short summary.

include informal whiteboard "mini-lectures" covering bite-sized concepts such as variables, loops, and functions, one at a time. These are interspersed with exercises from the textbook which are completed in pairs, sharing one laptop, as dictated by pair programming practices. Students have reported really liking this format; as one student commented in a course evaluation, *"This is a class that doesn't work well with a lecture style and she knows that and taught the class accordingly, small part lecture and then hands-on work."*

The second day each week focuses on circuits, handcraft, and digital fabrication as mediums for computational art and design. During these class periods, a document camera and equipment/materials are used to do live demos on handcraft techniques and also technical topics such as how to: use a multimeter, design a simple circuit (with a battery and a LED), prepare files for digital fabrication, and use CNC equipment. These demos are interspersed with long periods of unstructured hands-on work time, during which students are encouraged to move around, work in clusters, socialize, and ask the instructor or one another for help as needed.

Categories of student work and assessment are outlined in Figure 1, above. Assignments and quizzes focus on building core craft and programming competencies, while mini-projects and the final project offer the opportunity to integrate and apply these skill sets. (Mini-projects and the final project are detailed in the following section.) Homework includes readings from the textbook, work on mini and final projects, and sometimes finishing an in-class craft or programming exercise (although the bulk of this work happens *in* class).

B. Major Assignments

The major assignments of the course, mini-projects and the final project, emphasize blending handcraft with electronics, the creation of original vector designs in Processing, and realizing those designs in physical form. Although the majority of these assignments are structured around a piece of digital fabrication equipment, the emphasis is on what creative possibilities the equipment enables, rather than simply learning how to use it.

1) Mini-project #1: Felted Circuits: The first mini-project invites students to blend needle felting with textile circuitry, as inspired by the work of artist Moxie Lieberman [28]. Over a couple of class periods, students are taught how to needle-felt, how to design a simple circuit (with a light, a battery, and an LED), and considerations for working with electronic textile materials (like conductive sewing thread). Each student sketches a three-dimensional felted object, along with plans for how they will embed a sewn circuit into its structure – with care and attention given to LED placement, battery pack placement, and keeping the various threads from accidentally making contact and short-circuiting. Students who are interested in optionally adding a switch are encouraged to do so, and instructionally supported in modifying their sketches to accommodate this. Once students bring their sketches to the instructor for revisions and approval, they collect the necessary materials and bring their circuits to life. Examples of student work appear below, in Figure 2.

2) Mini-project #2: Plotter Drawings: The second mini-project invites students to blend code-driven/algorithmic design (done in Processing) with vector path *drawing*. Students are asked to create an original single-frame/non-animated vector design in Processing, which must also meet the following technical requirements:

- Use of 2+ drawing commands/shapes (`line`, `rect`, `ellipse`, etc.)
- Use of *variables* whenever possible.
- At least one *loop*, to create visual repetition.
- Appropriate use of comments throughout.

Students first submit a draft of their code to Moodle (our institution's LMS) and present their draft to the class for feedback using an overhead projector. While students are

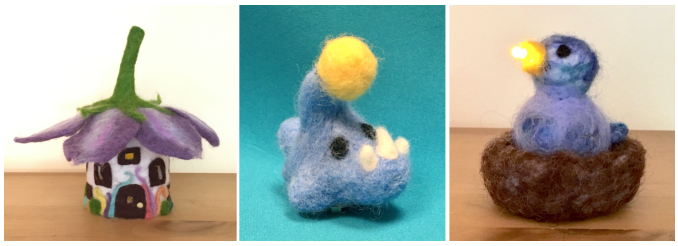

Fig. 2. Felted circuits designed by students: a fairy house that lights up inside when the flower is placed atop it, an angler fish, and a baby bird.

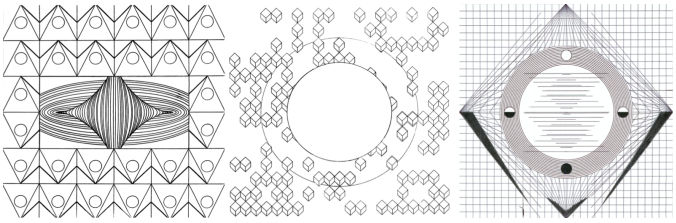

Fig. 3. Plotter designs produced by students as they learn to draw algorithmically. (Pen on paper, machine-drawn.)

working outside-of-class to incorporate any suggestions or revisions, class time is used to demonstrate how to export and format Processing PDFs for vector plotting and how to use a Cricut machine for drawing. Students submit a final draft of their code/design, output their design on the Cricut using pens or markers, and compose a written reflection about the experience. They present their final physical drawing in class alongside their code, again utilizing an overhead projector to do so. Examples of student work appear above, in Figure 3.

3) Mini-project #3: Vinyl-cut Stickers: The third mini-project invites students to blend code-driven/algorithmic design (done in Processing) with vector path *cutting*. This assignment follows the same draft-revision-fabrication structure as the plotter mini-project, but includes the added challenge of designing for cut paths instead of drawn lines; designing vinyl-cut stickers requires students to think about positive/negative space and open/closed shapes. Each term, an early observation and point of discussion is that overlapping lines in a design will generate vinyl confetti instead of a single, unified sticker.

Students must submit an original design that is significantly different in composition from their plotter design. While students are working outside-of-class to revise their sticker designs, class time is used to demonstrate how to work with a Roland vinyl cutter and how to carefully transfer a cut sticker to a surface. For their final presentations, students are required to show their sticker adhered to a surface; even a piece of paper suffices, but most students choose to affix their sticker to a bicycle, laptop, or other personally meaningful object. Examples of student work appear above at right, in Figure 4.

4) Final Project: The final project invites students to blend code-driven/algorithmic design (done in Processing) with handcraft, vector path drawing, vector path cutting, or a mix of these. Essentially, students are asked to generate a more complicated vector design than they have done prior and to realize it in any of the physical mediums covered in the course. They may also use any art/craft medium with which they have prior experience or wish to explore on their own.

This project follows the same draft-revision-fabrication structure as the two prior mini-projects, but with the added technical requirement that students must use *functions* in their code. Students are also expected to engage in independent research and planning in terms of creating a design that is suitable for their chosen medium(s), envisioning how they will realize this design in physical form, and requesting any necessary physical materials for their project. The fabrication part of this assignment must either incorporate handcraft in some way *or* demonstrate a more complicated application of a tool from an earlier assignment (for example, using a plotter with two pens/colors or using a laser cutter with a new material). Examples of student work are shown below, in Figure 5 – but students also used many other mediums such as charcoal, acrylic paint, and 3D printing.

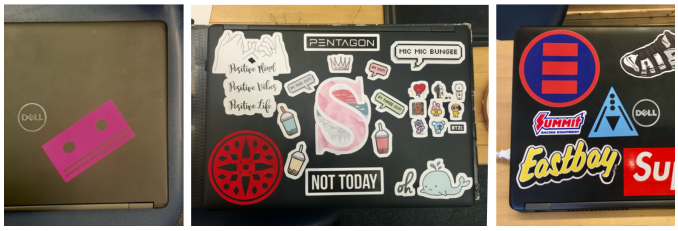

Fig. 4. Vinyl-cut stickers, which students valued enough to apply to their own laptops. (Student-produced stickers appear here in magenta, dark red, and sky blue, from left to right.)

5) Bonus Activity: Paper Circuits: Each term, one class period is spent teaching students how to make light-up Halloween cards (Fall term) or valentines (Spring term), as inspired by the work of Jie Qi [29], [30]. This activity is ungraded, as students take their cards with them at the end of class, but it is intended to reinforce earlier learning about circuits and to demonstrate yet another creative technical application.

During this class period, which always takes place after the felted circuit mini-project, students are taught how to create circuits on paper using copper tape, lights, and batteries. Students are also shown how to solder – and they have the

Fig. 5. Student final projects. Top row: laser cut coasters. Middle row: a graduation cap, start to finish. Bottom row: a hand-embroidered pillow, a two-color plotter drawing representing one student's African & Native American identity, an abstract hand-embroidered scene.

Fig. 6. Halloween-themed paper circuits.

opportunity to practice what they've already learned about using a multimeter to measure continuity. Some examples of student work appear above, in Figure 6.

6) Deprecated Assignments: In addition to the above, earlier offerings of the course included a textile sensor mini-project that followed the felted circuit mini-project. Inspired by Hannah Perner-Wilson's work [31], [32], this opened up discussion of resistance and the differences between a "dimmer" and a "switch". We used neoprene, Velostat, and conductive thread to follow Perner-Wilson's Instructable on the topic [33]. Students added their own flare to the assignment by making their sensors in creative shapes like dinosaurs and hearts – and enjoyed comparing how this and other variables affected their resistance, using a multimeter to investigate. Although students appreciated this more advanced electronics project, it was removed from later course offerings so that we could go more in-depth on frequently-requested, more advanced topics in computing, like translation and user interaction.

A couple of earlier terms of the course also used a laser cutter in lieu of a vinyl cutter for Mini-Project #3; this was simply dependent on whether we had the necessary equipment access, and the project requirements and challenges were comparable across the two different machines. During these offerings, students were also able to use the laser cutter for their final projects. Examples of student work from these deprecated assignments appears below, in Figure 7.

C. Infrastructure

The choice of using multiple infrastructure tools for teaching versus consolidating into a single platform has an effect on the tone of the class. Although there is a learning curve to students using multiple tools in tandem, this is common in the computer science workplace and we emphasize their utility with regard to preparation for post-graduation employment.

The course is taught entirely in Processing, aside from a Blockly-based warm up assignment. We chose Processing

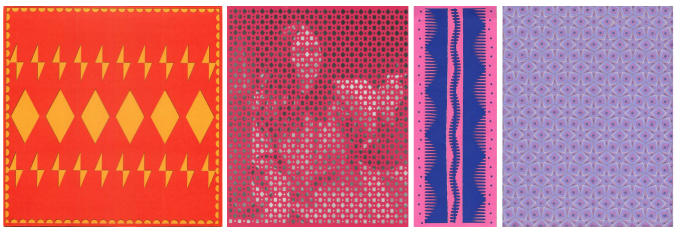

Fig. 7. Layered laser-cut paper, from an earlier version of Mini-Project #3.

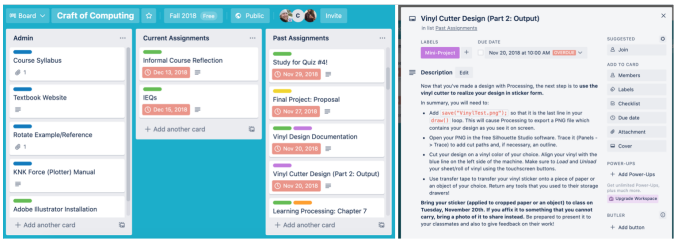

Fig. 8. The course website, as organized on Trello. An overview of current tasks and references (left) and a "card" offering guidance on the final phase of an individual mini-project (right).

because it was originally developed by artists and designers to enable the creation of creative work by those with minimal programming background [20]. To scaffold our journey through learning to write Processing code, all students are required to purchase *Learning Processing* and it serves as the course textbook [34]. *Craft of Computing* covers the first three chapters of the book in great detail, with select topics from later in the book covered by request.

A Trello board serves as our course website [35], [36], which colorfully organizes upcoming assignments/due dates, requirements for each assignment, and references such as the syllabus and equipment documentation. Students submit coursework on Moodle, except for physical artifacts which are submitted in person. An example from one of the course offerings is shown above in Figure 8.

Students are also required to join the *Craft of Computing* channel on our department's Slack team [37], which was created and is managed by our department's teaching assistants (TAs). This is where any announcements or clarifications are made in between course meetings, for example, deadline extensions or on-demand examples to clarify content. Students are also asked to post on Slack rather than emailing the instructor, as this enables a quicker response from the instructor, course TAs, or other students in the course. Students are also encouraged to send *direct* messages to the instructor and/or the TAs on Slack if they would like to inquire about grading, ask a question specific to their code, or anything more private.

Complementary to the above, Google docs is used to host software and equipment documentation and Calendly is used for students to schedule time on specific equipment [38].

D. Instructional Support

Teaching assistants have been absolutely instrumental in the success of *Craft of Computing*. Abundant teaching support is a unique benefit of teaching at a work college, but undergraduate tutors and graders could be tasked with similar support at other institutions. For this course, TAs staff evening lab hours and monitor the course Slack channel. They have also worked independently to create invaluable documentation that persists across terms; TAs have authored detailed step-by-step tutorials on each piece of equipment as well as file conversion processes. All of these feature annotated screenshots, lots of encouragement and a sense of humor. These walkthroughs have been vital in terms of reducing student confusion.

TAs also learn how to use each piece of equipment in advance of associated assignments and they supervise equipment time slots, which students can reserve both during and outside of evening lab. TAs also handle all of the signups and scheduling for this.

Finally, TAs grade all of the quizzes and meet with the instructor to grade final projects as a group at the end of each term. This is very helpful, as they have a window into each student's process and any barriers that they have encountered. On the whole, we have found it tremendously helpful to have our TAs' ongoing feedback on what topics students struggle with in lab and any issues that arise with equipment or materials.

V. Reflections

In this section, we summarize some of our own observations and reflections about the course.

A. Challenges & Rewards of Working in the Physical World

Working with physical materials and equipment can be time-consuming and frustrating, as echoed in students' course evaluations, especially as it contrasts with the software/digital focus of most computer science coursework. However, once students get in the habit of planning ahead and signing up for equipment time slots, they often express a sense of pride and triumph in what they make; a sentiment echoed through many student evaluations is, *"The more effort I put into a project, the more enjoyment I got out of it."* Students have especially enjoyed making things to display in their dorm rooms, to personalize their belongings, or to gift to loved ones – sometimes requesting items be returned early from grading in time for a birthday or holiday.

We have also observed students bonding over the above-mentioned frustrations; for example, how an earlier-used plotter would sometimes quit halfway through drawing or how awful the laser cutter smells after cutting wool felt. Students have also built a strong sense of community around our presentation days. Upon their suggestion, we stocked the lab with tea and second hand coffee mugs – and these class meetings came to be known as "CriTEAque Days". Sharing creative work is vulnerable and we believe that together we have cultivated a safe space for students to provide constructive feedback and try new things. This is echoed by a student evaluation: *"[The instructor] is very passionate about the topic and encourages us to try new techniques, or to apply these techniques differently to see the outcome."* Students look forward to our CriTEAque class periods, taking great interest in one another's creative and technical growth.

Students have also reported appreciation for learning craft techniques – as a creative outlet, as a destressor especially around midterms and finals, and as a means to be more self-sufficient. One student writes, *"I hadn't concerned myself prior with sewing or felting, but I have a fair deal of interest and respect for the creative applications of both after having taken the course."* Another reflects, *"This course taught me the basics of programming and basic sewing and embroidering techniques that not only I can use in a career but for life skills too."*

B. Importance of Growth Mindset at the CS0 Level

The importance of a growth mindset is well-documented, and the focus on creativity and craft in this course seems to support this objective. Most introductory computing courses have all students complete exactly the same assignments. This can offer the temptation to ask another student how they solved a bug, rather than struggling with it on one's own. The focus on creativity in this course means that every project is unique, and neither other students nor the instructor is likely to know the answers immediately. This allows abundant opportunity to practice debugging and growth mindset towards a goal the student is personally invested in. Student evaluation comments reflect this experience:

- *She was very relate-able and talked through struggles. She also did not just give us the answers to problems in our code. She just spotted the spots where they were and it was like hide and seek or find Waldo. It was great to hear the encouragement of knowing that she had found the bug in the code and then we had to work on finding them ourselves.*
- *She welcomes questions and mistakes on work and assignments and teaches us that sometimes mistakes aren't terrible things but can add on to our projects.*
- *The instructor does a good job of leading you into fixing code you have a problem with instead of outright telling you what is wrong or what you need to add so that you can actually learn something.*
- *Whenever students are stuck she just doesn't give them the answers she asks them questions for them to start "thinking like a computer scientist" as she would say.*
- *She has helped me to learn that coding is no harder than solving a puzzle.*

C. Leveling the Playing Field

CS0 courses strive to provide an entry point for students with little-to-no programming background, yet often enroll students with a wide range of preparation. For example, some students may opt to take CS0 as review or because they are interested in the specific topic/theme that is featured. This presents a unique challenge for CS0 instructors, as they strive to balance approachability for less experienced students with keeping more experienced students engaged.

Given our experience with this course, computational craft is, in fact, very well suited to this challenge. Between the creative aspect of every assignment, the variety of tools/materials/techniques at hand, and the extensibility of Processing as a programming language – we have seen students of all levels remain engaged over the course of each term. Student evaluation comments support this:

- *[The course] allows students to brainstorm and create their own unique projects while ensuring the students learn the content and the projects follow the specifications.*

- *... individuals in the class were at a variety of skills levels and [...] everyone was able to learn despite that challenge.*
- *Anytime that we finished a certain part, she would challenge us with extra tasks that gave a better understanding.*

D. Preparation for CS1

CS0 courses aim not only to offer an appropriate entry point to computer science, but also to prepare students with minimal programming background for CS1. Student evaluations reflect success in these areas as well:

- *Even though I had no background of Computer Science, [...] this course [was] very accessible to me.*
- *I recognize that this would be an ideal first computer science course.* (This comment was made by a student with some programming background already.)
- *I would recommend anyone who is thinking about doing computer science to take this course. It's a perfect preparatory course for [CS1] and [CS2] by softly introducing key concepts that are crucial to the major. Anyone who takes this course will have a huge leg up in [CS1].*

Despite the non-traditional computing topic area which may be perceived by some as "softer" or less difficult, *Craft of Computing* is one of the only CS0 courses in our department to use a text-based programming language; most CS0 courses in our funnel utilize block languages. A couple of students posited in interviews that this makes for a smoother transition to our CS1 course taught in Python; *Craft of Computing* students already have familiarity with compiler errors, nuances of written syntax, and data representation (e.g. ints vs. floats). Our coverage of Processing also uniquely exposes students to debugging, libraries, and programming in different coordinate systems.

E. Broadening Participation & Perceptions through Craft

The goals we were most passionate about for this course were to invite participation from a diverse cross-section of students and to vastly broaden students' perceptions of what computer science is "good for".

A total of 69 students enrolled in *Craft of Computing* over the two-year interval, many of which were first year students. At our institution, first year students are placed into Fall Term courses by an advisor, while all students self-enroll for the Spring Term. Looking only at Spring Term (self-enrolled) students, 19 men and 15 women enrolled in the course. While a small course like this is hardly suitable for reporting statistics, this is 44% female enrollment, as compared to under 20% of computer science bachelor's degrees being awarded to women nationwide [39].

We are also very encouraged that student evaluations reflect success in broadening perceptions of computing:

- *Every assignment is flexible; the criteria can be met with an incredible number of solutions of varying complexity, and every assignment feels like it has the potential to be an art project.*
- *... the course did well to link the computational and physical areas, generally broadening the scope for which I might consider programming.*
- *I learned so incredibly much! Before this course, I didn't know the first thing about programming, but now, I'm coding simple video games in my free time, and I have even decided to minor in computer science. I loved this course so much that I applied for, and was granted the opportunity to work as one of the two TAs for the course in the fall.*
- *... her ability to relate the computing information to things in the real world, and to combine digital work with analog work is simply astounding.*
- *This was a great opportunity to learn new things in a new field. I liked being able to bring my code to art.*

VI. Recommendations

In this section, we make concrete recommendations to others who may be interested in offering a similar course at their own institution.

A. Course Logistics

Longer course periods really do offer more time to engage with physical materials, fabrication equipment, and handcraft. They also allow for more meaningful discussions, both on the topic of readings and on days that students are presenting their work. We recommend scheduling a course of this type during extended time blocks, if your institution offers this option.

The earliest offerings of the course were taught in a couple of different classrooms that did not have storage for materials or student projects – nor did they have the equipment used in the class. The instructor spent a lot of time shuffling both materials and students between locations, to ensure that we had access to everything we needed. If at all possible, we recommend scheduling a course of this type in a lab space – ideally one in which students feel at home.

In terms of resources, students adored *Learning Processing* as a textbook. They appreciated the author's conversational tone and the workbook-style exercises, which we leveraged both for homework and in class. Students had no trouble skipping ahead to specific topics of interest on their own if they were looking for a challenge; in fact, one of the most common pieces of feedback received regarding the textbook was simply a desire to have covered more of it. Some students also independently sought out the textbook author's accompanying videos which explain and demonstrate key concepts, and told us how helpful these were. In short, a textbook and related videos that *directly support student activities* worked well.

Students initially complained a bit about all of the infrastructural pieces (Moodle, Trello, Slack, Google docs, etc.). We emphasized Trello and Slack as the most important resources to keep track of, encouraged students to configure Slack notifications to their phone or email, and took great care to appropriately link between platforms. Ultimately, many students did find Slack to be a helpful touchstone throughout the term. We believe messaging on Slack feels closer to a text

message than an email, and that this allows students to very quickly get in touch without worry over formality, etiquette, or perfect phrasing. Thus, we do recommend using Slack – or another platform emphasizing approachability, to facilitate announcements and peer support between course meetings.

B. Equipment & Materials

As mentioned in our earlier reflections, working with physical tools and materials is uniquely challenging – especially within a computer science context, where students are used to having everything they need to complete assignments right on their laptops. Over time, we have found a few strategies that helped to ease this.

We let students know in the course description, and again on the first day of class, that this course will require visiting the evening lab and/or scheduling separately with equipment. This helps to set student expectations early on and to redirect students who may have an incompatible term schedule.

We also make sure to offer plenty of unstructured time to work on projects during course meetings. This is especially helpful when learning about circuits, doing any kind of handcraft, and when students are pursuing open-ended final projects. Because of the demands of students' labor schedules, this helps to ensure that students have the necessary time, support, and access (to equipment, materials, and instructional staff) to achieve course goals. Depending on your institutional context, you may choose to do this as well.

To keep the class on schedule, we recommend purchasing any necessary equipment and materials before the start of the term. The one exception to this is students' final project materials, which students request through a Google spreadsheet. If your department does not have funding available to cover these needs, you may consider charging a materials fee for the course.

When selecting equipment, simple is best. Early course offerings incorporated a laser cutter and vinyl cutter housed in a neighboring department, plus a powerful-but-experimental plotter. By the time of the course's latest offering, we had scaled back to mostly relying upon a Cricut machine (which can both cut and draw). The Cricut software is also easier for students to learn and to stick with over multiple assignments, especially compared with learning one application for a laser cutter and another for a vinyl cutter.

C. File Formatting & Exporting

It can be confusing for students to envision how the display output of their coded designs will translate into the physical world. For each Processing-based mini-project, it is important to be clear that the goal is generation of static images – not animations – because only vector data will be used to generate machine output. Students are welcome to embellish their code with color and fills on their shapes, but it is important to emphasize that only the line data will be used.

Although Processing does allow for exporting vector designs to PDF, some additional formatting is necessary. Most notably, Processing exports duplicate paths for each shape: one to represent lines and one to represent fill, even when `noFill()` is specified. The first time an assignment requires formatting student designs for a piece of equipment, we recommend doing a live demo of the steps required to achieve this. We include a quick overview of vector versus raster file formats and a brief-but-targeted dip into Adobe Illustrator.

Our TAs have assisted with creating written documentation of this process for students to follow along with on their own. In addition, we provide some simple example files for students to practice this process on. We recommend ensuring that you have tested the entire workflow and that your students have access to any necessary software – even one lab computer with vector graphics software (e.g. Adobe Illustrator) installed will suffice. A dedicated lab computer can also be very useful for running the digital fabrication equipment, rather than having students "print" to this equipment from their laptops.

D. Emphasizing Original Creative Work

From the course's inception, we had intended for students to create original designs for every assignment, although we were open-minded about what that meant. However, some students dedicated quite a lot of time to recreating familiar imagery with Processing, while technically meeting each assignment's coding requirements. In some cases, this meant spending hours plotting out an existing logo or character coordinate-by-coordinate, failing to leverage the built-in Processing functions we had wanted students to learn about – and missing out on the joy and creativity of algorithmic design.

In later terms, we added an explicit requirement that assignments consist of original creative work rather than anything derivative. This yielded better progress toward programming learning goals and also more interesting outcomes. Showcasing beautiful examples of prior student work, as they were accumulated, helped greatly with this as well.

Finally, from a student course evaluation: *"I suggest she buy more needle threaders."* This is indeed a great thing to keep in mind when teaching with textiles!

VII. Conclusion

In designing *Craft of Computing*, we had hoped to further broaden participation within our own undergraduate department, and to expand students' perceptions of computer science. Anecdotal evidence suggests that the course was effective in doing so; the course enrolled students across a diversity of majors (including art, theater, and applied design) and students report a broader understanding of the field in their course evaluations. The work that students produced – as exemplified in this paper – is uniquely creative and personally meaningful, piquing students' interests in the creative and varied possibilities of computing.

We hope that sharing our experiences and recommendations emphasizes the importance of diversified CS0 offerings and can, in particular, enable more courses of this variety at other institutions.

REFERENCES

[1] Z. J. Wood, J. Clements, Z. Peterson, D. Janzen, H. Smith, M. Haungs, J. Workman, J. Bellardo, and B. DeBruhl, "Mixed approaches to CS0: Exploring topic and pedagogy variance after six years of CS0," in *Proceedings of the 49th ACM Technical Symposium on Computer Science Education*, 2018, pp. 20–25.

[2] M. Brown, "CS0 as an indicator of student risk for failure to complete a degree in computing," *Journal of Computing Sciences in Colleges*, vol. 28, no. 5, pp. 9–16, 2013, publisher: Consortium for Computing Sciences in Colleges.

[3] J. Pearce and M. Nakazawa, "The funnel that grew our CIS major in the CS desert," in *Proceedings of the 39th SIGCSE Technical Symposium on Computer Science Education*, 2008, pp. 503–507.

[4] M. Haungs, C. Clark, J. Clements, and D. Janzen, "Improving first-year success and retention through interest-based CS0 courses," in *Proceedings of the 43rd ACM Technical Symposium on Computer Science Education*, 2012, pp. 589–594.

[5] L. Buechley and M. Eisenberg, "The LilyPad Arduino: Toward wearable engineering for everyone," *IEEE Pervasive Computing*, vol. 7, no. 2, pp. 12–15, 2008, publisher: IEEE.

[6] Y. B. Kafai, E. Lee, K. Searle, D. Fields, E. Kaplan, and D. Lui, "A crafts-oriented approach to computing in high school: Introducing computational concepts, practices, and perspectives with electronic textiles," *ACM Transactions on Computing Education (TOCE)*, vol. 14, no. 1, pp. 1–20, 2014, publisher: ACM New York, NY, USA.

[7] S. Cheryan, A. Master, and A. N. Meltzoff, "Cultural stereotypes as gatekeepers: Increasing girls' interest in computer science and engineering by diversifying stereotypes," *Frontiers in Psychology*, vol. 6, p. 49, 2015, publisher: Frontiers.

[8] A. Master, S. Cheryan, and A. N. Meltzoff, "Computing whether she belongs: Stereotypes undermine girls' interest and sense of belonging in computer science." *Journal of Educational Psychology*, vol. 108, no. 3, p. 424, 2016, publisher: American Psychological Association.

[9] J. Margolis and A. Fisher, *Unlocking the Clubhouse: Women in Computing*. MIT Press, Feb. 2003.

[10] J. Margolis, *Stuck in the Shallow End: Education, Race, and Computing*. MIT Press, Feb. 2010.

[11] C. Lewis, P. Bruno, J. Raygoza, and J. Wang, "Alignment of goals and perceptions of computing predicts students' sense of belonging in computing," in *Proceedings of the 2019 ACM Conference on International Computing Education Research*, 2019, pp. 11–19.

[12] N. Veilleux, R. Bates, C. Allendoerfer, D. Jones, J. Crawford, and T. Floyd Smith, "The relationship between belonging and ability in computer science," in *Proceeding of the 44th ACM Technical Symposium on Computer Science Education*, 2013, pp. 65–70.

[13] E. Höhne and L. Zander, "Belonging uncertainty as predictor of dropout intentions among first-semester students of the computer sciences," *Zeitschrift für Erziehungswissenschaft*, vol. 22, no. 5, pp. 1099–1119, 2019, publisher: Springer.

[14] J. Morales-Chicas, M. Castillo, I. Bernal, P. Ramos, and B. L. Guzman, "Computing with relevance and purpose: A review of culturally relevant education in computing." *International Journal of Multicultural Education*, vol. 21, no. 1, pp. 125–155, 2019, publisher: ERIC.

[15] E. M. Lovell, "A Soft Circuit Curriculum to Promote Technological Self-Efficacy," Master's thesis, Massachusetts Institute of Technology, 2011.

[16] Q. Cutts, E. Cutts, S. Draper, P. O'Donnell, and P. Saffrey, "Manipulating mindset to positively influence introductory programming performance," in *Proceedings of the 41st ACM Technical Symposium on Computer Science Education*, 2010, pp. 431–435.

[17] L. Murphy and L. Thomas, "Dangers of a fixed mindset: implications of self-theories research for computer science education," in *Proceedings of the 13th Annual Conference on Innovation and Technology in Computer Science Education*, 2008, pp. 271–275.

[18] B. Simon, B. Hanks, L. Murphy, S. Fitzgerald, R. McCauley, L. Thomas, and C. Zander, "Saying isn't necessarily believing: influencing self-theories in computing," in *Proceedings of the Fourth International Workshop on Computing Education Research*, 2008, pp. 173–184.

[19] S. Cheryan, V. C. Plaut, P. G. Davies, and C. M. Steele, "Ambient belonging: how stereotypical cues impact gender participation in computer science." *Journal of Personality and Social Psychology*, vol. 97, no. 6, p. 1045, 2009, publisher: American Psychological Association.

[20] C. Reas and B. Fry, "Processing: Programming for the media arts," *AI & SOCIETY*, vol. 20, no. 4, pp. 526–538, Sep. 2006.

[21] I. Greenberg, *Processing: creative coding and computational art*. Apress, 2007.

[22] C. Reas and B. Fry, *Processing: a programming handbook for visual designers and artists*, 2nd ed. Cambridge, Massachusetts: The MIT Press, 2014.

[23] O. Astrachan, T. Barnes, D. D. Garcia, J. Paul, B. Simon, and L. Snyder, "CS principles: piloting a new course at national scale," in *Proceedings of the 42nd ACM Technical Symposium on Computer Science Education*, 2011, pp. 397–398.

[24] D. Xu, U. Wolz, D. Kumar, and I. Greenburg, "Updating introductory computer science with creative computation," in *Proceedings of the 49th ACM Technical Symposium on Computer Science Education*, 2018, pp. 167–172.

[25] L. Buechley, M. Eisenberg, and N. Elumeze, "Towards a curriculum for electronic textiles in the high school classroom," in *Proceedings of the 12th Annual SIGCSE Conference on Innovation and Technology in Computer Science Education*, 2007, pp. 28–32.

[26] G. Jayathirtha and Y. B. Kafai, "Electronic textiles in computer science education: a synthesis of efforts to broaden participation, increase interest, and deepen learning," in *Proceedings of the 50th ACM Technical Symposium on Computer Science Education*, 2019, pp. 713–719.

[27] "High-Low Tech." [Online]. Available: http://highlowtech.org/

[28] Moxie, *I Felt Awesome: tips and tricks for 35+ needle-poked projects*. Cincinnati, Ohio: North Light Books, 2010.

[29] J. Qi, "The fine art of electronics: paper-based circuits for creative expression," Master's thesis, Massachusetts Institute of Technology, 2012.

[30] J. Qi and L. Buechley, "Sketching in circuits: designing and building electronics on paper," in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, 2014, pp. 1713–1722.

[31] H. Perner-Wilson and L. Buechley, "Making textile sensors from scratch," in *Proceedings of the fourth international conference on Tangible, embedded, and embodied interaction*, 2010, pp. 349–352.

[32] Kobakant, "How to Get What You Want." [Online]. Available: https://www.kobakant.at/DIY/

[33] H. Perner-Wilson, "Instructables | Plusea (Hannah Perner-Wilson)." [Online]. Available: https://www.instructables.com/member/Plusea/

[34] D. Shiffman, *Learning Processing: a beginner's guide to programming images, animation, and interaction*. Morgan Kaufmann, 2009.

[35] "Trello." [Online]. Available: https://trello.com/

[36] "Craft of Computing (Course Website)." [Online]. Available: https://trello.com/b/WGeaxo5n/craft-of-computing

[37] "Slack." [Online]. Available: https://slack.com/

[38] "Calendly." [Online]. Available: https://calendly.com/

[39] National Center for Science and Engineering Statistics, "Women, Minorities, and Persons with Disabilities in Science and Engineering: 2021." National Science Foundation, Alexandria, VA, Tech. Rep., 2021. [Online]. Available: https://ncses.nsf.gov/wmpd