Computational Art - Introducing High School Students to Computing via Art

Zoë J. Wood
Computer Science
California Polytechnic State University
San Luis Obispo, CA 93407
zwood@calpoly.edu

Paul Muhl
Santa Barbara High School
700 E. Anapamu St.
Santa Barbara, CA
pmuhl@sbunified.org

Katelyn Hicks
Computer Science
California Polytechnic State University
San Luis Obispo, CA 93407
khicks@calpoly.edu

ABSTRACT

Introducing computer science to high school students in a creative context, and fitting such a course into an overly packed high school curriculum, is a challenge. This paper describes a fruitful collaboration to create and teach an alternative introductory computing curriculum for high school students targeted at introducing students to computing via an artistic context. This curriculum allows students to be introduced to computing in a highly interest based context, attracting diverse student body participation (25% female students in the first offering and 38% female students in the present offering). This curriculum also fulfills student's fine art course requirement making it more readily accessible to a wide audience to 'try out' CS while making progress on their academic goals. This curriculum has been used as a successful introduction to computing at the Santa Barbara High School CS Academy in the 2014-2015 school year and is currently being offered again this year. In this paper, we present the curriculum and evaluation of this initial offering.

Keywords

Introductory programming, computer science education, applied computing - fine arts, k-12 education

1. INTRODUCTION

Context based introduction to computer science has been consistently shown to be a positive way to introduce computing concepts [11, 8, 9, 6]. Presenting computer science within a context helps students to recognize the importance of computing to a field or theme they are interested in and creates a greater sense of investment in the course material and projects. Numerous themes have been proposed and various universities have adopted context based introductory courses.

Nationwide, computational art (closely related to media computation and generative art), is a relatively under-utilized introductory computing context, however this context has been shown to be successful at various institutions [7, 4, 10], with recent work by Guzdial [10] presenting long term analysis of the success of the Media Computation course at Georgia Tech.

SIGCSE '16, March 02 - 05, 2016, Memphis, TN, USA
© 2016 Copyright held by the owner/author(s). Publication rights licensed to ACM. ISBN 978-1-4503-3685-7/16/03...$15.00
DOI: http://dx.doi.org/10.1145/2839509.2844614

Meanwhile, high schools have been tackling the problem of introducing computing to a wide audience in an attractive context. In addition, high schools face the challenge of an already impacted school day with numerous required courses, leaving students with little opportunities to explore computing as an elective course. This kind of exploratory enrollment is essential for students to be able to try out computer science to discover the joys of problem solving and programming.

This paper addresses the need for high school curriculum that allows students to fulfill course requirements and build an interest and understanding of computing in a creative and interesting context. Building off of a University level introduction to computing course focused on computational art, this paper describes a collaboration focused on developing a yearlong high school computational art course. This course simultaneously introduces students to computer science and to art and design principles in a hands-on creative manner, allowing students to fulfill their fine arts requirement while also teaching them to program in Java.

In this paper, we present the computational art curriculum and initial evaluation detailing the successes of the year long high school course. This course was attractive to a wide audience and received overwhelming positive responses for both shifting student's identities positively towards computing and encouraging them to enroll in subsequent computer science courses.

1.1 Context

Santa Barbara High School (SBHS) is committed to computer science education for all high school students and started teaching computer science again in 2012 after a break from offering such classes, followed by a School Board approved Computer Science Academy in 2014. The Academy operates as a school within a school serving those students who self identify as interested in the computer science (examples of other academies at the same school include Visual Arts and Design and Media Arts and Design Academies). This high school of about 2200 students and includes a diverse student body with 56% Latino, 49% low socioeconomic status and 55% first generation college bound students. The CS Academy offers four courses in computing, Exploring CS, AP CS, Mobile computing and Computational Art.

The Computational Art course was designed as a collaborative effort with Professor Wood in order to offer an alternative entry-way into computer science. This course attracts a diverse student body; the initial offering in 2014-2015 included 25% female students (versus the Exploring CS course with 10% female students) and the 2015-2016 course has 38% female students. Taught by the CS Academy Director, Paul Muhl, this course allows students to fulfill their fine arts requirement[1] while learning to program in Java. Of all the courses offered by the CS Academy, this course best reflects the general student population and is an attractive entry to computer science for students of any grade level.

The collaboration on this high school level course comes out of the success of the introductory computational art course, which has been taught at Cal Poly for the last four years as a CS0 course. The course material explores the relationship of aesthetics and computation. Students write text based computer programs that generate two-dimensional digital art using the Processing development environment. The course allows students to explore the joy of using mathematics to define shapes, curves, relationships, color and animations to create artistic expressions. No prior programming experience is assumed. This course is one of five context based courses offered as a CS0 course for majors and has been shown to be successful at preparing students for subsequent computer science courses. Other context options include mobile, security, robotics and video games. The computational art offering has attracted larger than its share of female students then the other CS0 courses offered at Cal Poly (assuming an even distribution of female students across all contexts).

1.2 The Basics of the Curriculum

The purpose of the SBHS Computational Art course is to introduce, broaden and enhance a student's ability to generate both static and dynamic digital art (static art is most equivalent to an image, while dynamic art includes animation or changes based on user interaction). This course fulfills student's fine arts requirement and therefore has a solid grounding in art and design principles. The medium used to introduce these principles is the Processing development environment, which allows them to simultaneously develop their artistic and programming skills. Processing is an open source programming language and IDE, which is built on the Java programming language. It was designed by Casey Raes and Benjamin Fry at the MIT Media Lab and was built to allow for the teaching of fundamental programming concepts in a visual context.

2. RELATED WORKS

There is a large body of work addressing the multitude of challenges (including failure rates [1]) in introductory computer science curricula, including context-based computing, small courses and the use of labs early, making use of pair programming, etc. [9, 6, 12, 5, 3]. In the high school setting, all of these challenges exist in addition to the challenge of fitting computer science into an already packed curriculum.

Previous research has suggested that computational art can serve as a socially relevant, contextualized, real-world context for the introduction of computer science concepts. For example, Greenburg, Kumar & Ku evaluated the success of an art-focused creative-coding environment designed to emphasize excitement, creativity, and innovation within computer science and found that their "approach is successful and appears particularly appealing to women" [7]. Data suggested that major and non-major students taking the computational art course were more "positively inclined to take additional CS courses," and "more likely to spend extra time on a homework assignment for 'fun'." Guzdial [10], in a review of research over 10 years focused on efforts to teach introductory computer science courses with a media-focused context, found that the introduction of this Media Computation course improved retention.

Qualitative analysis suggested that retention was linked to students experiences in the course as "a welcome opportunity to be creative," and as contextualized and relevant [10]. Of special relevance, analysis indicates that not only did more women enroll in these courses than other introduction to CS courses, but more women succeeded - in some cases at a higher rate than men. While more research is necessary to fully account for the factors that lead to these higher interest and higher success rates for women in these classes, these findings make intuitive sense given women's higher participation in AP art courses. In 2013 women made up 75% of AP exam takers in AP Studio Art: Drawing; 74% of AP exam takers in Studio Art: 2D Design; 71% of AP exam takers in Studio Art: 3D Design; and 66% of AP exam takers in Art History [2]. These numbers indicates a fruitful area for creating compelling computer science courses, appealing to female students in particular.

3. OUR SOLUTION

In order to address the need for a creative and engaging high school course of interest to a wide audience of participants, we designed the computational art course to allow students to learn and apply key art and design principles and key technical programming principles to the production of both static and dynamic digital art.

In the computational art course, students gain an understanding of digital media production, including the production of such art via text based programming. Students' understanding comes from viewing, discussing and creating their own digital media projects. Students study fundamentals of digital art (color, shape, composition, relationships, perspective, texture, light, and animation principles and timing). Through the study of these fundamentals, students create multiple artistic pieces demonstrating an understanding of these principles. Students also develop their own personal style for the creation of both static digital art pieces and dynamic, interactive or animation visual art projects.

Students simultaneously develop their technical understanding of key technical concepts relating to computational art production including data representations, control flow, iteration and the application of logic to control the interaction and story of their creations, in addition to mastering the appropriate Java programming syntax. Students then apply this knowledge to implement computer programs that produce artistic artifacts.

The course is broken into six units, each of which includes material focused on art and design concepts. Students engage with the concepts by learning from existing examples

[1]Specifically for the context of this high school, students in California applying to UC schools are required to complete a year of fine arts

and then mastering relevant technical skills and material to produce computational art demonstrating the particular concept.

The six fundamental units include:

- Principles of design, color, shape, and space
- Principles of design, composition, scale and perspective
- Principles of Design: generative & repetition and organic shapes & curves
- Principles of design: texture and light
- Digital Image representation and manipulation
- Principles of animation

The primary learning modality for the course beyond introduction, demonstration, and exploration of each concept included associated assignments to be completed by students in the computer lab with the instructors assistance for each of these units. This hands-on learning allows students to create computational art demonstrating their understanding of the relevant art and design concept as well as their mastery of the technical material.

Each of the unit topics are enumerated in detail below. Each unit contains subtopics related to the global principle and example assignments and the associated key technical concepts (related to programming) that are covered with each topic and assignment. The learning outcomes for each unit are encapsulated in the subtopics and closely related technical skills, with each outcome enumerated and expanded upon for each individual assignment. For each subtopic the curriculum is designed such that students will learn and understand the principles for each subtopic and technical skill and be able to apply that knowledge via the production of an assignment that illustrates their use. Example assignments for each subtopic and the related technical skills are briefly enumerated here to illustrate the copasetic relationship between both the art and design principles and the programming principles being introduced to the students.

The order in which the art and design topics are introduced has some flexibility, however, the units are currently designed in this specific order to allow students to scaffold their technical skills in such a way to build more and more complex computer programs.

3.1 Principles of Design, Color, Shape, and Space

This first unit allows students to begin to develop their art and design vocabulary along with introducing them to the development environment and programming syntax. See Figure 1 for an example of some of the student's art from the 2014-2015 offering of the computational art course at SBHS related to this unit.

- **Subtopic:** Shapes and order

  Assignment 2D pixel space: design a face
  Technical Introduction to the Processing development environment and syntax

- **Subtopic:** Shapes, order, and color

  Assignment Overlapping shape composition: design a creature
  Technical order of commands, color picker and shape syntax

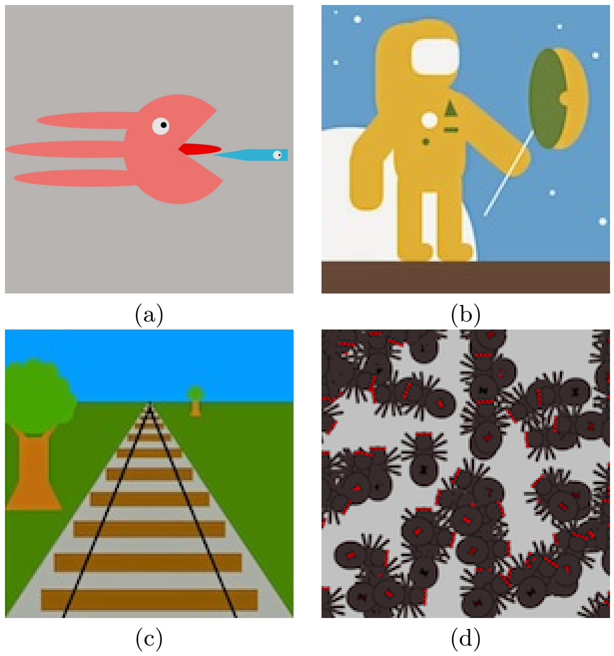

Figure 1: Student work from the 2014-15, assignments focused on: (a) creatively understanding shape and 2D space (pair programmed E. Ornelas and D. Panchi), (b) an assignment exploring composition through the study, reproduction and modification of a Blexbolex design (E. Diaz) (c) exploring perspective and the application of scale (E. Heffernan) (d) exploring repetition and generative art creation (E. Rojas)

- **Subtopic:** Mastering 2D space and ordering

  Assignment Growing and shrinking dots
  Technical the use of variables to control size

- **Subtopic:** Color (warm/cool and representation)

  Assignment Warm/cool lab
  Technical the use of variables to control color

3.2 Principles of Design, Composition, Scale and Perspective

This second unit builds on student's vocabulary and art and design concepts and allows students to practice their basic programming skills. See Figures 1 and 2 for examples of static art produced for this unit.

- **Subtopic:** Introduction to composition

  Assignment Golden ratio lab
  Technical variable practice

- **Subtopic:** Composition, scale and mood

  Assignment Composition exploration - Blexbolex
  Technical introduction to transforms to rescale and reposition a character

- **Subtopic:** Perspective and vanishing points

  Assignment Vanishing point lab
  Technical practice with transforms

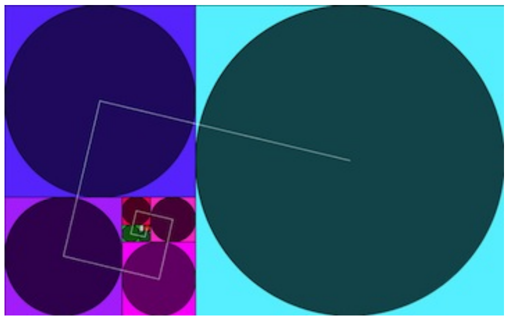
Figure 2: Student result for an assignment practicing using variables to control a sketch that conforms to the golden ratio and practices art and design concepts of composition and scale (A. McKaba)

3.3 Principles of Design: Generative & Repetition and Organic Shapes & Curves

This third unit allows students to explore the differences between generative art using random numbers and repition and careful design. This unit also introduces the use of mathematical functions (parametric curves) in order to allow students to create organic shapes. In this unit students enhance their programming skills to add more control flow tools to apply repetition to the creation of their art pieces. See Figure 1 for some examples of static art produced for this unit.

- **Subtopic** Generative art, iteration and random #s

  Assignment Exploring repetition - random repetition
  Technical loops
- **Subtopic** More generative art

  Assignment Further repetition: face with many teeth
- **Subtopic** Organic shape: Parametric functions for lines, curves and circles

  Assignment Creating flowers
  Technical loops (emphasis on loop control parameters)
- **Subtopic** Organic shape continued

  Assignment Creating spirals and other natural forms
  Technical loops and mathematical background

3.4 Principles of Design: Texture and Light

This fourth unit introduces fundamental art and design concepts relating to mood via light and texture. Students expand and apply their understanding of loops to include nested loops in order to create painterly renderings. See Figure 3 for some examples of students work.

- **Subtopic** Texture

  Assignment Comparative textures lab (lines, points, strokes and hash marks)
  Technical nested loops and variables for strokes
- **Subtopic** Light and form

  Assignment Difference between a circle and sphere
  Technical loops
- **Subtopic** Light, form and texture

  Assignment Parametric curves and stroke textures
  Technical loops and mathematical background
- **Subtopic** Light, form and texture

  Assignment Implicit curves and stroke textures
  Technical loops and mathematical background

3.5 Digital Image Representation & Manipulation

This fifth unit introduces students to the digital representation of images. This unit encourages students to take self portraits with their cell phones and then modify them programmatically in creative ways. It includes an examination of pop-art such as Andy Warhol and combines creativity and technology in a compelling way for students to practice and deepen their understanding of data representations and loops. Examples can be seen in Figure 3.

- **Subtopic** Introduction to images

  Assignment Painterly strokes applied to images
  Technical 2D arrays and indexing (loops)
- **Subtopic** Pop-art and Images

  Assignment Andy Warhol filter
  Technical 2D arrays and conditionals
- **Subtopic** Images changing over time

  Assignment Image processing: morph
  Technical 2D arrays and indexing (loops)
- **Subtopic** Images: 2D arrays, indexing

  Assignment Image processing: edge detection
  Technical 2D arrays and indexing (loops)
- **Subtopic** Images and shapes together

  Assignment Images with windows defined by shape
  Technical 2D arrays and 2D space conditionals

3.6 Principles of Animation

This final unit introduces a few of the principles of animation [13] in order to help students create compelling animations. Some more advanced technical skills, such as objects, are introduced to allow students to see more powerful datastructures in action. This unit includes a capstone experience for the course in the form of team based final animation that tells a story.

- **Subtopic** Introduction to motion and time (history of animation and keyframing)

  Assignment Animation loop
- **Subtopic** Animation timing basics: frame rate and velocity

  Assignment Make a character move/respond
  Technical interaction with the mouse
- **Subtopic** Procedural animation

  Assignment Make many moving creatures
  Technical array practice
- **Subtopic** Principals of animation: squash and stretch, staging and story

  Assignment Curve following with squash and stretch

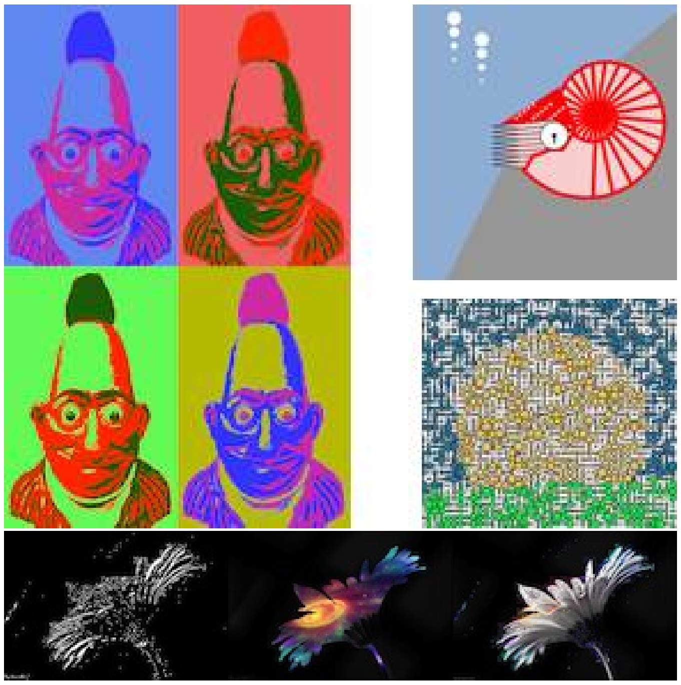

Figure 3: Further samples of student work created with this curriculum. On the left top row is an example of an Andy Warhol filter (E. Diaz). On the right top is an example of an assignment exploring the use of parametric curves to create organic shapes and designs (A. McKaba), while below this is a 'painterly' sketch with the use of defining various shapes to represent strokes is applied to a scene defined by implicit curves, allowing students to explore texture (B. Saguache). Bottom row, example student's work from an assignment focused on digital image representation including the implementation of simple edge detection and compositing (A. Gaona, D. Ocampo, A. McKaba).

- **Subtopic** Intro. to physically based animation: particles

  Assignment Fireworks
  Technical Introduction to objects

- **Subtopic** Intro. to hierarchical modeling for animation

  Assignment Build a duck
  Technical Use of the matrix stack

- **Subtopic** Hierarchical modeling for animation II

  Assignment Make the duck walk and fly
  Technical Use of the matrix stack

- **Capstone** Final Project: An original animation

4. EVALUATION AND REFLECTION

Overall, this course has been a successfully collaboration to create an engaging context based introduction to computing. We are very satisfied with the curriculum and with student's engagement with the course.

4.1 Survey results

At the conclusion of the yearlong course, twenty-eight students responded to a survey about the course. Of those students responding to the survey, when asked "How interested are you in learning more programming/computer science?", twenty responded that they were either "very interested" (i.e. planning to enroll in a CS related course the following year) or "possibly interested" (considering enrolling in a CS related course). Only two students responded that they were "not interested" in taking future CS courses (with the final 6 responding that they 'might try one more course' at some future time). The fact that 71% of the students who took this course are interested in taking more courses indicates that the course was a success at inspiring student's interest in computer science.

Students were asked at the beginning and end of the course, what they identified as: technologist, artist, both, or neither. When asked what they identified as at the beginning of the year, nine students identified as technologists, twelve as artists, five as both, and two as neither. When asked to reevaluate their identity at the end of the year, eight identified as technologists, five as artists, fourteen as both, and one as neither. Nine students began the year identifying as only a technologist or only an artist, but now identify as both a technologist and an artist. We see this switch in identity as a very positive outcome for the course as one of the targets of the course is to expose a wide variety of students to the joys of computer science, even those who perhaps initially only identified with the context domain of art.

We asked students to list their top three favorite assignments and their final animation assignment (creating a simple story) was an overwhelming favorite. With the next most popular assignments being an animation assignment focused on simulating fireworks using a particle system and an image processing assignment including both edge detection and image composition (see Figure 3).

One-way to measure the students' engagement is to ask them about extra time and which assignments they showed to others. The top assignment in both of these categories was the final animation assignment – again indicating a strong correlation in student satisfaction with this capstone experience of being able to create and code a story programmatically. The other assignments that students told their friends and family about were "Images - morph" and "Images - Andy Warhol", again indicating studentsâĂŹ enthusiasm for being able to work programmatically with the more complex data structure.

4.2 Instructor Reflections

Overall, the course worked extremely well. Though the coding component started slowly in favor of an early emphasis on art principles, by the end of the course more computer science topics were covered in more depth than would be required or expected at this class level. The end result is that the students moving up to the AP course from Computational Art are very well prepared, with an early exposure to concepts such as object oriented programming and some advanced data structures. Furthermore, a wide variety of math concepts were introduced and used above and beyond what some of the students had yet learned in their math courses. These math concepts were made accessible to all students in the class either through explanation and derivation or simply as "black-box" formulas whose effects could be understood through experimenting with input parameters to see immediate, visual results. Again, the end result was a clear demonstration of the close relationship between math and programming, and a confidence to use code in a modular way. The Animation unit was the only one that proved to be a bit too difficult, coming at the end of the year, and representing the full synthesis of all the most complicated concepts. Students were still able to achieve a basic level of success with the first sub-topics of the unit and the final animation project, but in retrospect this unit will need to be presented differently in the future.

4.3 Examples of student work

At the conclusion of the year, the instructor and several students from the course participated in a district wide art show, demonstrating the students work to a public audience and receiving positive feedback from the audience in response to the student's art in addition to an "innovation award" for the student art.

Throughout the year students produced a wide variety of both static and dynamic art pieces shown in Figures 1, 2 and 3. We include static figures of sample student work, which is just a subset of the lab assignments to provide examples of the type of work created by students.

5. CONCLUSIONS

This preliminary offering of computational art at the high school level was a great success. Of the computer science course offerings at SBHS, it had and has the most diverse student population. The collaborators were able to restructure the college level curriculum to an appropriate level for the high school students and expand the curriculum to more clearly address the related art and design principles. This enhanced emphasis on art and design skills also allowed the course to fulfill student's fine art requirement, making it accessible for a wide audience of students to 'try out' CS. Surveys show that the course overwhelming inspired students to pursue learning more about computer science and that taking the course shifted students' identities positively towards computer science. The fact that the second year of the course offering has an even larger female student population, 38% female, is likewise a huge indicator of the course's success. We plan to make the entire curriculum publicly available in the next year and hope that other high schools are inspired to use this highly successful course in their curriculums.

6. ACKNOWLEDGMENTS

We wish to thank all the 2014-2015 SBHS computational art students for their enthusiasm and art.

7. REFERENCES

[1] J. Bennedsen and M. E. Caspersen. Failure rates in introductory programming. *SIGCSE Bull. 39*, pages 32–36, 2007.

[2] C. Board. The 10th annual AP report to the nation - subject supplement. http://apreport.collegeboard.org/, 2014.

[3] K. E. Boyer, R. S. Dwight, C. S. Miller, C. D. Raubenheimer, M. F. Stallmann, and M. A. Vouk. A case for smaller class size with integrated lab for introductory computer science. In *Proceedings of the 38th SIGCSE Technical Symposium on Computer Science Education*, SIGCSE '07, pages 341–345, New York, NY, USA, 2007. ACM.

[4] R. Bryant, R. Weiss, G. Orr, and K. Yerion. Using the context of algorithmic art to change attitudes in introductory programming. *J. Comput. Sci. Coll.*, 27(1):112–119, Oct. 2011.

[5] J. P. Cohoon and L. A. Tychonievich. Analysis of a CS1 approach for attracting diverse and inexperienced students to computing majors. In *Proceedings of the 42Nd ACM Technical Symposium on Computer Science Education*, SIGCSE '11, pages 165–170, New York, NY, USA, 2011. ACM.

[6] Z. Dodds, R. Libeskind-Hadas, C. Alvarado, and G. Kuenning. Evaluating a breadth-first CS1 for scientists. In *ACM SIGCSE Bulletin*, volume 40, pages 266–270. ACM, 2008.

[7] I. Greenberg, D. Kumar, and D. Xu. Creative coding and visual portfolios for CS1. In *Proceedings of the 43rd ACM Technical Symposium on Computer Science Education*, SIGCSE '12, pages 247–252, New York, NY, USA, 2012. ACM.

[8] M. Guzdial. Teaching computing for everyone. *J. Comput. Sci. Coll.*, 21(4):6–6, Apr. 2006.

[9] M. Guzdial. Does contextualized computing education help? *ACM Inroads*, 1(4):4–6, Dec. 2010.

[10] M. Guzdial. Exploring hypotheses about media computation. In *Proceedings of the Ninth Annual International ACM Conference on International Computing Education Research*, ICER '13, pages 19–26, New York, NY, USA, 2013. ACM.

[11] M. Haungs, C. Clark, J. Clements, and D. Janzen. Improving first-year success and retention through interest-based CS0 courses. In *Proceedings of the ACM Technical Symposium on Computer Science Education*, 2012.

[12] N. Nagappan, L. Williams, M. Ferzli, E. Wiebe, K. Yang, C. Miller, and S. Balik. Improving the CS1 experience with pair programming. *SIGCSE Bull.*, 35(1):359–362, Jan. 2003.

[13] F. Thomas and Johnston. *Disney Animation - the illusion of life.* Abbeville Press, New York, NY, USA, 1981.