A Mathematics and Digital Art Course

Vincent J. Matsko
University of San Francisco

vince.matsko@gmail.com

Abstract

In the Fall 2016 semester, a Mathematics and Digital Art course was taught at the University of San Francisco for the first time. After a brief overview, course content for the first half of the semester is presented chronologically. Then the use of Processing and student Final Projects is discussed, as these two course components comprised the second half of the course. The intent is to give someone interested in teaching such a course a sense of what it would look like in practice. Several examples of student work are also presented.

Overview of the Course

I would like to share my experience with designing and implementing a digital art course for those interested in developing a similar course. The implementation is presented roughly chronologically in order to give a sense of what teaching such a course would look like on a week-by-week basis.

In September 2015, I wrote a blog on creativity in mathematics [4]. In addition to blogging about puzzles and problems, I starting writing about how to create digital art. Later on that Fall semester, when the department was making a tentative course schedule for the 2016–2017 academic year, I thought: why not take things one step further and design a course about digital art?

Because my work involved mathematics and coding very heavily, it would not be difficult to make sure the course had substantial mathematical content – students would not simply be using drop-down menus in pre-existing image-processing software, but working with code to generate digital images. Here is the course description I wrote for Mathematics and Digital Art:

> What is digital art? It is easy to make a digital *image,* but what gives it artistic value? This question will be explored in a practical, hands-on way by having students learn how to create their own digital images and movies in a laboratory-style classroom. We will focus on the Sage/Python environment, and learn to use Processing as well. There will be an emphasis on using the computer to create various types of fractal images. No previous programming experience is necessary.

Broadly speaking, the course was a digital art laboratory. The class was held in computer lab, so students were creating images almost every day. Students explored most topics in the Sage environment [6] – classes usually began with a brief presentation followed by work on the computer. All the Sage worksheets are available on the course website [5].

I should point out a few of the advantages of my particular circumstances, as they directly impacted the quality of the course. First, the course was offered as part of our First-Year Seminar program, where courses are kept small so students get to know one particular instructor fairly well. Second, there was a First-Year Seminar Assistant program, so I was able to have a student assistant in class each day who earned course credit for helping me. My assistant (Nick) knew Python and was a double major in mathematics and art – and also presented at Bridges 2016 – so was an ideal candidate for the position. His assistance was invaluable. Although we had nine students, we might have easily worked with 20–25 students. And third, I was able to have my class in a computer laboratory, which made demonstrations and in-class work especially easy.

Sequence of Topics for the First Half of the Semester

First, we discussed color. My inspiration was *Interaction of Color* by Josef Albers [1]; his work is accessible to beginners. We explored color in relation to other colors as he did – but this required learning hexadecimal numbers and RGB values, which most students had heard about but never really understood. I did decide to work exclusively in the RGB system since we were not making digital prints. (I would like to include printing in the future, but this requires resources not currently available.)

Students created images as shown in Figure 1. The pieces can be thought of as being composed of squares subdivided into rectangles; in the left image in Figure 1, for example, there are four squares divided into two rectangles each. The centers of these rectangles are the same color, and the two border colors are such that their RGB values average to produce the center color. For example, if the border colors were yellow and red, the center color would be orange since

$$\frac{1}{2}((1,1,0)+(1,0,0)) = (1,0.5,0).$$

Of course this is not how Albers created his images, since he used pigment – but this method mimics his style, however simplistically.

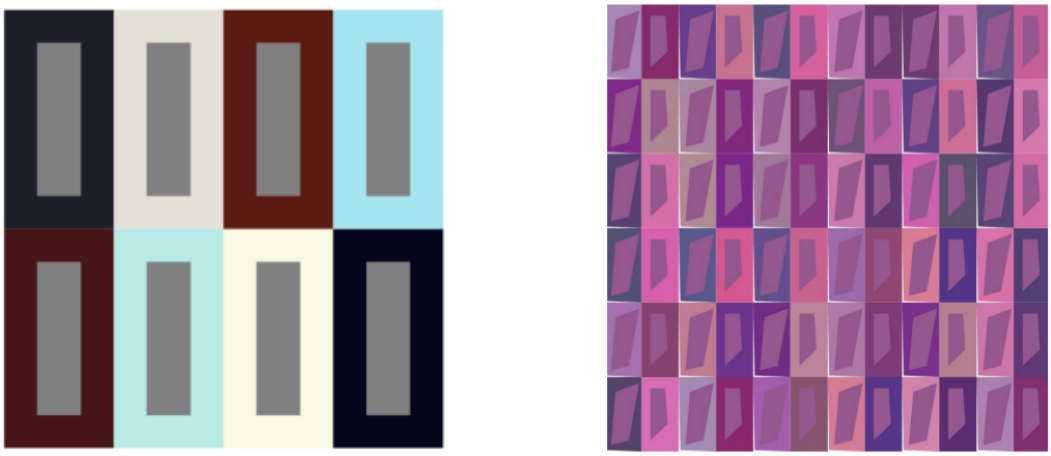

Figure 1 : *Josef Albers: Andrew's image (left) and Ella's image (right).*

It was encouraging to see students approach this assignment in ways I had not anticipated. Andrew took a minimalistic approach, while Ella explored breaking the symmetry of the component squares.

I should remark that this assignment (and most others) included a brief narrative discussing the student's choice of parameters. Of course some choices are just a matter of preference (like color choice, perhaps), but some are more deliberate. It is important that students begin to articulate these thoughts. In addition, since most assignments involved randomness in some way, the same parameters could produce widely differing images depending on the seed used for Python's random number generator. So one aspect of many assignments was generating several different images with different random number seeds, and then choosing the image which was most aesthetically pleasing.

Next, we looked at various ways randomness can be used to create different visual effects. I wrote a Sage worksheet which produced an array of circles whose colors and radii could be randomly generated. For example, if the radius parameter is given by

$$0.5 + 0.5 * \texttt{random}(),$$

no circle will have a radius smaller than 0.5, and the largest circles will have radii close to 1. Similar ideas can be used to randomly create a range of colors. The larger the coefficient of the `random()` function, the greater the parameter range.

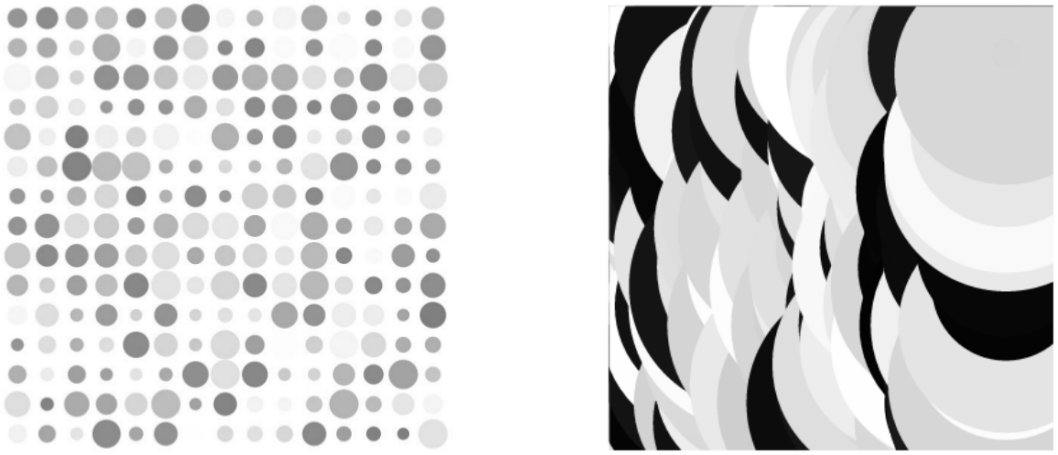

Figure 2 : *Textures: Adrianna's image (left) and Maddie's image (right).*

It was interesting to see the wide range of images the class created. Notice how some students, like Adrianna, avoided having the circles interact all – although her use of the light gray gives the piece a sense of motion. On the other hand, some students, like Maddie, took advantage of the drawing algorithm (rendering the circles from left to right, bottom to top) to create a scalloped effect by using larger circles.

We then moved on to working with color gradients. The basis for this topic was one of the first digital pieces I ever created, *Evaporation* (see Figure 3). In this piece, the top of the image is one color, and randomness is gradually added to the RGB values as you move down the image. Randomness can be added more or less quickly near the top. In Figure 3, if the top of the image is assigned $y = 0$ and the bottom of the image is assigned $y = 1$, then the randomness added to the color of a circle centered at (x, y) is proportional to y^2. By altering the exponent of y, different color gradients may be produced. Moreover, randomness is also incorporated by having the circles be of different radii, very much like the textures described above.

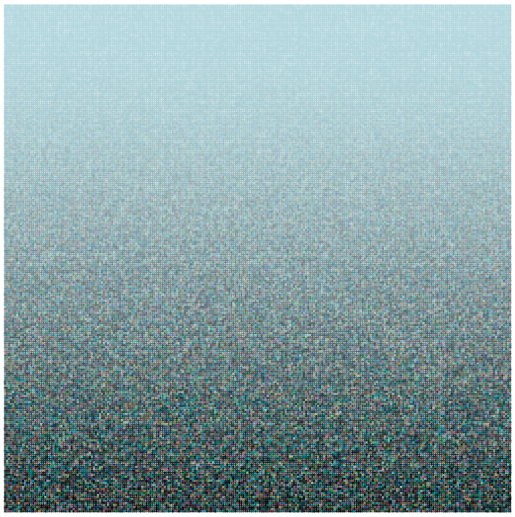

Figure 3 : *Evaporation.*

The effect of the randomness of the circle sizes is subtle in *Evaporation.* Students, however, used this randomness to great effect, as shown in Figure 4. Notice how an initial small radius with a large randomness coefficient, as in Maddie's image, creates a very different effect than a small randomness coefficient, as seen in Julia's image.

The geometry of linear and affine transformations was next. This was an important discussion, since affine transformations played a rather large part in the course. They served as a basis for a study of fractals generated by iterated function systems of affine transformations. Beginning with the Sierpinski triangle, the idea was to show how self-similarity could be described by means of affine transformations, and then use

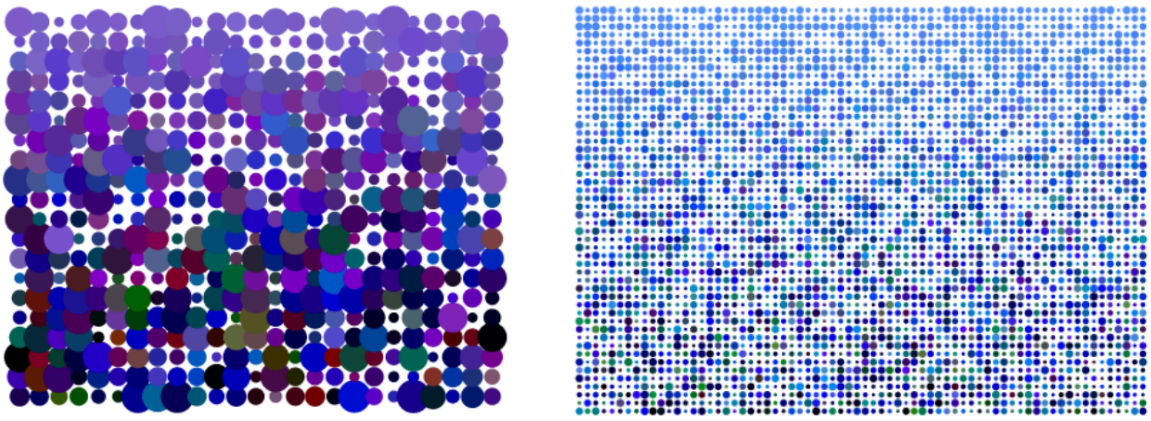

Figure 4 : *Color gradient: Maddie's image (left) and Julia's image (right).*

those transformations to generate an image of the fractal using a Sage worksheet. The algorithm was the usual iterative one of creating a sequence of points by randomly choosing transformations.

This was challenging for many students, but I designed an interactive Sage worksheet which showed the effects of any affine transformation on a unit square. Further, not only did students need to describe the effect of an affine transformation on a unit square, but they also had to write the affine transformation given a unit square and the parallelogram it is transformed into. There was a strong emphasis on giving the algebra of matrices a geometric interpretation.

Since students were very familiar with the Sierpinski triangle, I gave an assignment which let them explore it further. The prompt for this assignment was to create an image which was simultaneously as close *and* as far from a Sierpinski triangle as possible. In other words, the image should be identifiable as being based on a Sierpinski triangle – but just barely. There were many interesting submissions; see Figure 5.

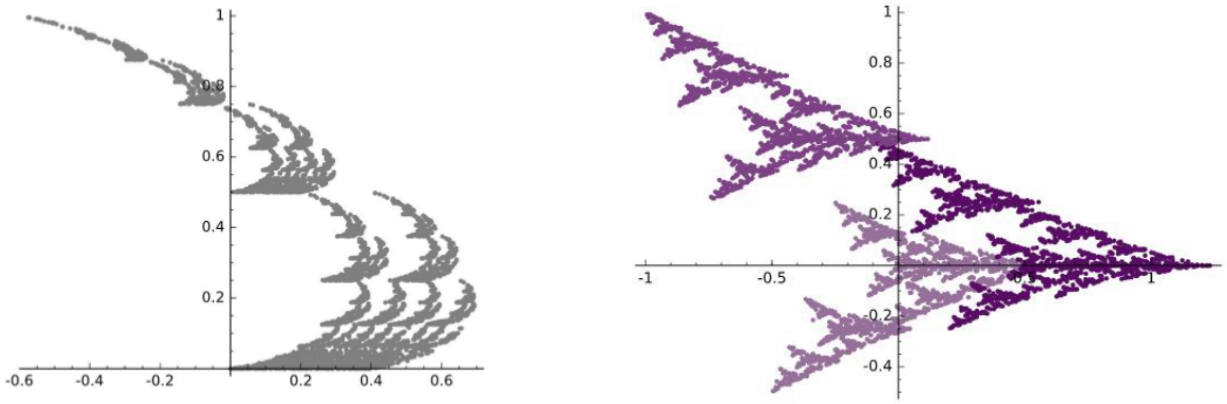

Figure 5 : *Sierpinski triangle variation: Adrianna's image (left), and Safina's image (right).*

The motivation behind this assignment was to have students think more carefully about the affine transformations used in an iterated function system. This assignment required a bit more thought than one where students just randomly typed in affine transformations and looked at the fractals produced.

A two-week unit on polyhedra was next. This did not relate to the course, really, but I thought it would be good for students to get some exposure to three-dimensional geometry. However, students remarked on this apparent disjointedness in their written comments, and so I intend to replace this unit with one on L-systems. An excellent source for L-systems is [3], which may freely be downloaded as a pdf file.

Processing

Getting this far took roughly nine weeks. The rest of the semester had two emphases: learning Processing, and undertaking Final Projects. Each week, we would spend Monday and Wednesday working with Processing, and then students would work on their Final Projects on Friday.

I used the Python mode in Processing, and started students out with simple movies I had created and had them alter the code to produce different effects. They were interested in the code itself, so I discussed some of the Processing files one line at a time so they knew what each function call did.

It is not practical to give examples of movies here; there are examples of student work on Day069 of my blog [4]. The main programming idea used throughout was linear interpolation. Essentially, a movie is generated as a sequence of individual frames. If a movie had 300 frames (about ten seconds) and you wanted the background of the first frame to be black and the background of the last frame to be white, you could interpolate between the colors $(0, 0, 0)$ and $(1, 1, 1)$ to create the effect of the background becoming lighter and lighter shades of gray. In fact, any feature which is described with a numerical parameter may be animated using linear interpolation, such as the size of a circle, its location on the screen, and so on. Students were given assignments which specifically required the use of linear interpolation in different ways.

For in-class work, I made short movies and had students duplicate them exactly. These movies incorporated linear interpolation in various ways, so making copies of the movies required students to describe the linear interpolation mathematically. This was challenging for many students, but in the end, all were successful.

Finally, the Processing functions `mouseX` and `mouseY` return the x- and y-values, respectively, of the mouse on the Processing screen. This allows for the creation of effects depending on the location of the mouse, so the user can interact with the movie. Students really enjoyed this feature of Processing, and many were very creative with its use. Some students used Processing in their Final Projects, although this was not required.

Final Projects

As mentioned earlier, interwoven with the days creating movies in Processing was work on Final Projects. About midway through the semester, after they had seen several different ways of creating digital images, students needed to choose a Final Project to work on. This was a very open-ended assignment – I wanted to give students as much room for creativity as possible.

I had asked then to submit a project Proposal, but this turned out to be a bit too frustrating. For most students, the Project only took a final form after three or four weeks of work – they really did need to experiment and see what directions might prove fruitful. So in future semesters, I do not plan to have a formal Proposal assignment, but rather informally discuss plans with students individually. As the small class size allows me to talk with each student every class period, I can easily monitor their progress without the need for a written Proposal.

As much as possible, I worked to accommodate students' ideas. Two students wanted to work with image processing – but the difficulty was that image processing in the Sage environment (which sends computations to a remote server to be evaluated and then returned) was not feasible due to the length of time it took worksheets to be evaluated.

So Nick (my seminar assistant) took these two students under his wing. This involved researching image processing libraries in Python, downloading them, getting them to work properly on the students' laptops, and making the routines accessible to the students. Not an easy task! Without an assistant, I would not have been able to give these students the attention they needed to be successful with their projects.

Madison (see Figure 6) wanted to experiment with impressionism. We had worked a lot with circles and color, and she wanted to process photographs (which she had taken herself) by replacing pixels with small circles. She found that a gray background allowed the colors of the individual circles to stand out, and worked with circle size to create an impressionistic effect while still maintaining a sense of realism (so that the original image was still discernible).

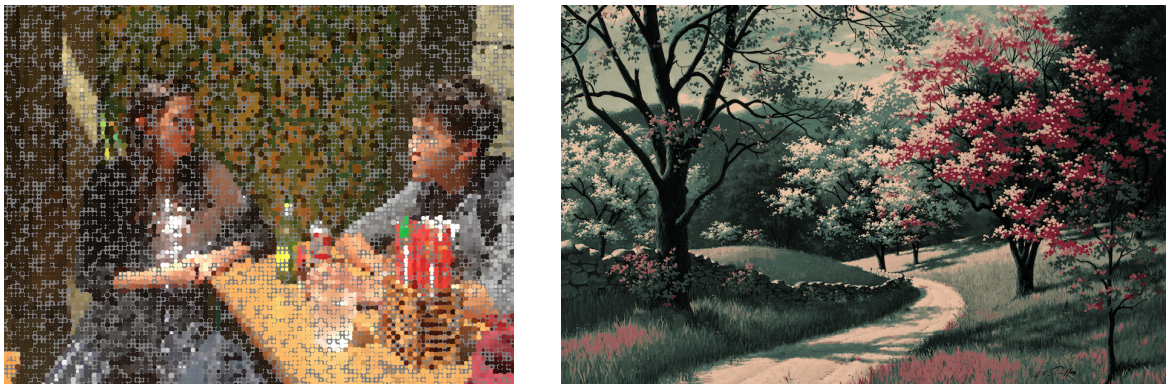

Figure 6 : *Image processing: Madison's image (left) and Lucas's image (right).*

Lucas wanted to experiment with color adjustments. He first found various color palettes involving just a few colors. Then for each pixel in the image, he found the color in a given palette closest to it (using the usual Euclidean distance formula on the RGB values) and assigned the pixel that color and gave it the same intensity as in the original image.

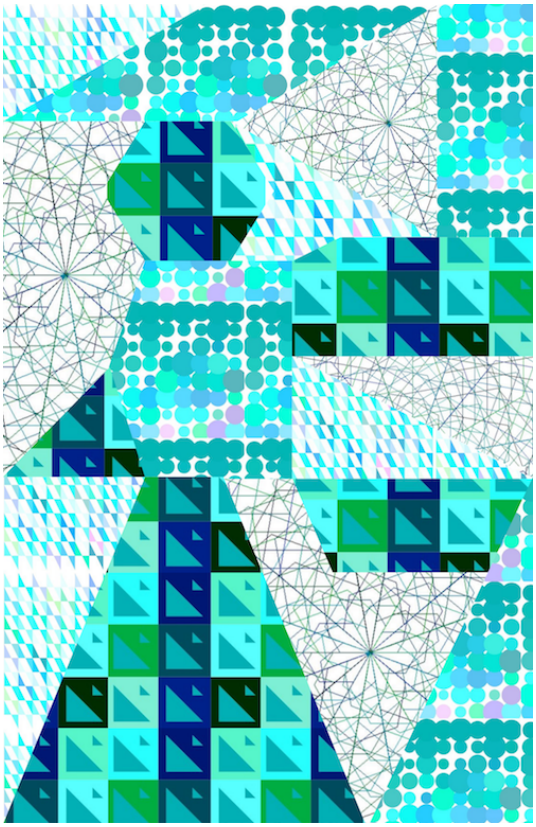

Figure 7 : *Safina's Final Project.*

Safina wanted to incorporate all of the important ideas in the course into her project. Work with gradients, color, and Albers-like ideas can all be seen in Figure 7. But Safina also wanted to add something new, and she ended up researching how to use turtle graphics in Python. The regions with the spidery filaments were created this way.

Andrew and Julia were both interested in Josef Albers (see Figure 8), although they evidently created their own geometry for their color experiments and layered their colors differently. Note that the central circles in Julia's image are both the same color.

One of Sharon's friends was interested in the work of Salvador Dali, so she decided to incorporate his work into her project. We did not work with realism of any kind in the course, so Sharon decided to focus on an aspect of Dali's work she could easily work with: color.

She looked for pieces which had backgrounds which were basically two-colored. Then, she worked at recreating the color pattern by combining two color gradients, one on top of the other (see Figure 9).

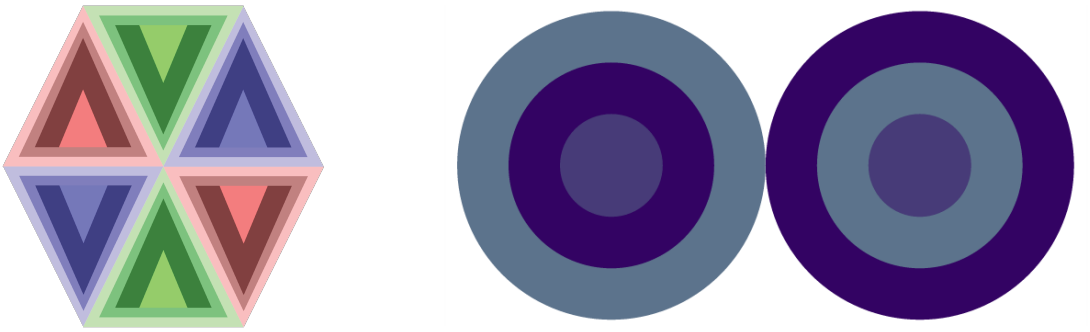

Figure 8 : *Josef Albers: Andrew's Final Project (left) and Julia's Final Project (right).*

Ella was interested in L-systems, and so I wrote some routines for her to use in her Final Project (and which I will also use when I now teach L-systems in the course). She was able to create some interesting effects by just slightly altering the parameters to L-systems which created trees and superimposing the new L-systems on top of the original. This gave some depth to the trees in her forest. A still from one of her movies is shown in Figure 10.

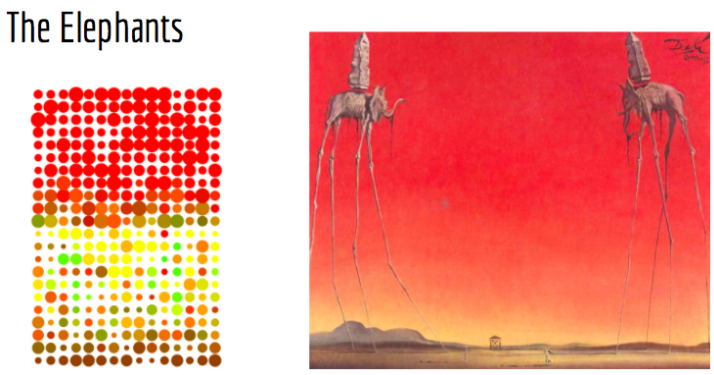

Figure 9 : *Sharon's Final Project.*

Students also wrote a final response paper about how they felt the course went, and how their attitudes about mathematics, art, and computer science changed over the course of the semester. I will let the quotes speak for themselves.

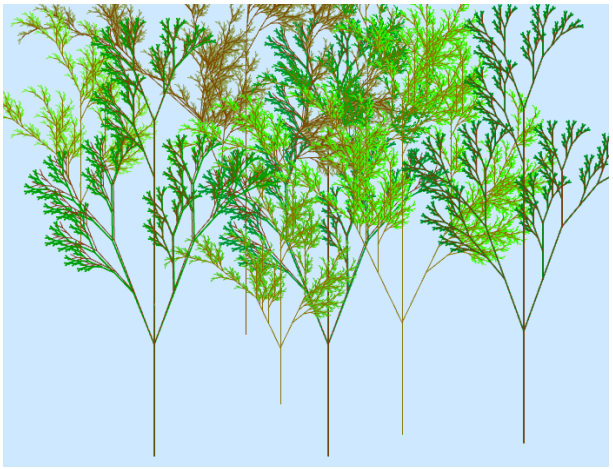

Figure 10 : *Ella's Final Project.*

"After this course, I definitely think about math differently, because now I know how it can be used to figure out shapes and layers and colors that I can use in my art. I also think differently about art, because before this course, I had only really done traditional art, and had no idea about any digital art besides using a tablet to draw with instead of a pencil. This course has really opened my mind to what I think art can be, and definitely how it can be created in different ways."

"Best of all, this class is part of the reason why I decided to declare a minor in computer science. It is something I have been considering as I have always had an interest in the subject, but I feel this class had really helped fuel that interest and give me the final nudge I needed."

Other Course Features

There were two additional features of the course which need to be mentioned. First, in addition to the presentation for their Final Project, students also gave two presentations on papers from The Bridges Archive [2]. This is a repository of papers presented at Bridges conferences since 1998. Students could choose a six- or eight-page paper on a topic of their choice; it did not have to relate directly to the course. This turned out to be a successful assignment – in their written comments, a few students singled these weeks out as particular favorites. These presentations took place in the sixth and twelfth weeks of the course.

Because of its place in the First-Year Seminar program, Mathematics and Digital Art came with a small budget. As a result, I was able to invite three local mathematical artists to speak: Chamberlain Fong, Carlo Sequin, and Shirley Yap. They spoke on very different topics, and were popular with the students.

Implementing a Similar Course

Finally, I would like to remark that I have tried to make the course website [5] as complete as possible so others interested in teaching such a course might do so without needing to start from scratch. And certainly my selection of topics and assignments reflects my own personal interests.

I do not use a text for the course, but rather have written several blog posts with material relevant to the course. For ease of reference, I will list those posts here. For a post on Josef Albers, see Day002 [4]; for L-systems, see Day007–009; for textures and color gradients, see Day011–012; for iterated function systems, see Day034–036; for tutorials on Processing, see Day 039–044; and for a guided tour through the Fall 2016 semester as it was happening (including more examples of student work), see Day057, Day059, Day061, Day063, Day066, Day069, Day071, and Day072. (To see Day039, for example, go to the url `cre8math.com/tag/Day039`.)

Acknowledgments

I would like to thank all my students from my Fall 2016 Mathematics and Digital Art class! They were an inspiration, and generously granted me permission to use their work in this paper. Also, I would especially like to acknowledge my course assistant, Nick Mendler, who contributed so much to the success of the course.

References

[1] Albers, Josef, *Interaction of Color,* Yale University Press, New Haven, 2006.

[2] Bridges Organization, The, *The Bridges Archive,* `http://archive.bridgesmathart.org` (as of Jan. 26, 2017).

[3] Lindenmayer, A., and Prusinkiewicz, P., *The Algorithmic Beauty of Plants,* Springer-Verlag, New York, 1990.

[4] Matsko, Vincent J., *Creativity and Mathematics.* `http://www.cre8math.com` (as of Jan. 26, 2017).

[5] Matsko, Vincent J., *Mathematics and Digital Art, Fall 2016.* `http://vincematsko.com/Fall2016/MAT195/2016F_DigArt.html` (as of Jan. 26, 2017).

[6] SageMathCloud. `https://cloud.sagemath.com/settings` (as of Jan. 26, 2017).

[7] Wikipedia: The Free Encyclopedia, *L-system.* `https://en.wikipedia.org/wiki/L-system` (as of Jan. 26, 2017).