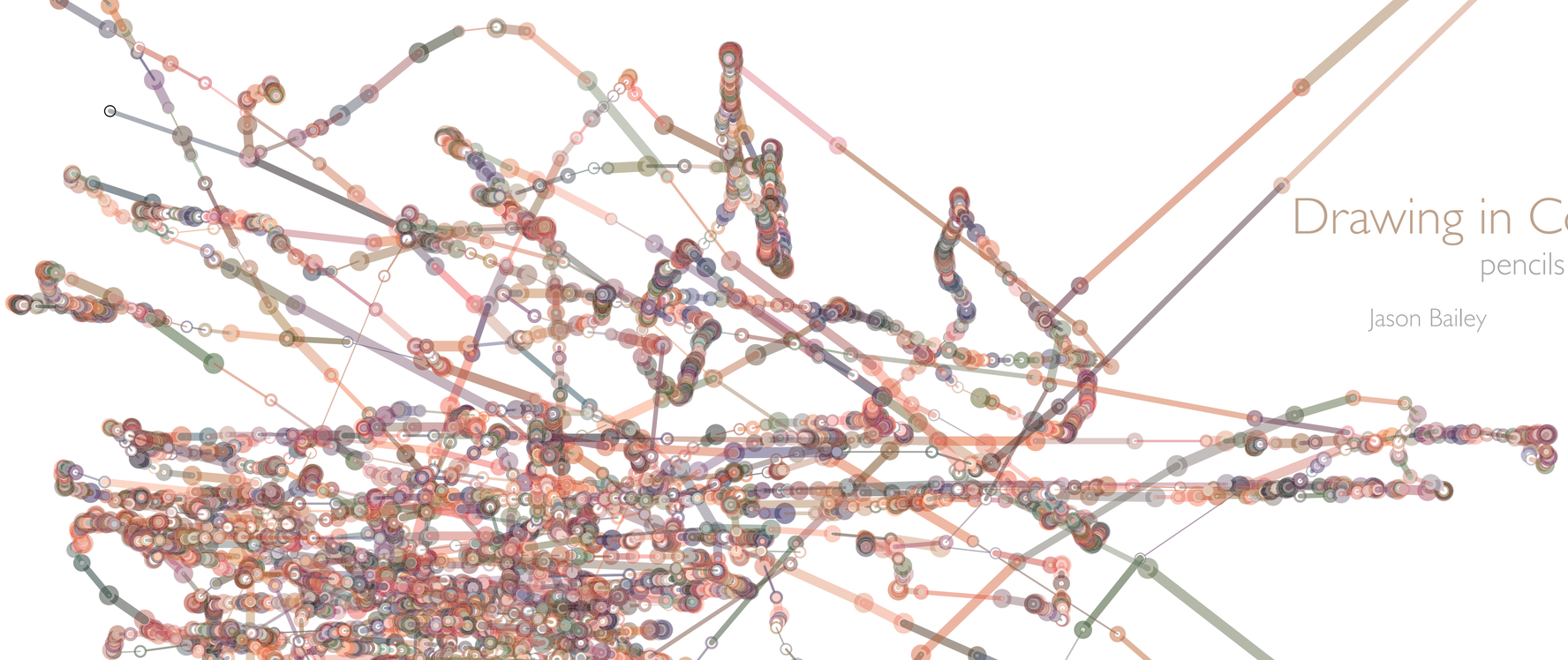


Drawing in Code

pencils pixels process

Jason Bailey

Drawing in Code

pencils pixels process

Jason Bailey

This thesis is submitted in partial fulfillment of requirements for
the degree of Master of Fine Arts in Design and approved by MFA Design
Review Board of Massachusetts College of Art and Design in Boston.

May 2010

Jan Kubasiewicz, Thesis Advisor
Coordinator of Graduate Program in Design, Professor of Design
Dynamic Media Institute,
Massachusetts College of Art and Design, Boston

Joseph A. Quackenbush,
Associate Professor of Design
Dynamic Media Institute,
Massachusetts College of Art and Design, Boston

Contents

"We must expect great innovations to transform the entire technique of the arts, thereby affecting artistic invention itself and perhaps even bringing about an amazing change in our very notion of art."

Paul Valéry,
Pièces sur L'Art, 1931

Acknowledgements

Jan Kubasiewicz
For listening, encouraging and challenging me, and for convincing me that my ideas are worth following.

Joseph Quackenbush
For helping me find my own voice in writing and encouraging me to use it to tell stories.

Erin
For telling me to quit my job and go back to school, and for your
patience with me during the challenges of the last three years.

My DMI Family
For inspiring, humbling, and educating me and expanding
my understanding of the world.

Thank You!

Introduction

I have spent my whole life using drawing as a means of understanding and clarifying my environment, synthesizing that understanding, and sharing it with others.

Sometimes drawing has meant trying to accurately recreate what I see around me, such as creating a still life of some interesting objects on a table or a portrait of a family member. This experience provided me an opportunity to examine my world in much closer detail than I otherwise do in daily life.

Drawing has also meant taking highly technical information, like the complex physics involved in a car accident, and boiling it down into an easily understood graphic so that others might understand it better; or illustrating statistical data, like the variable temperature in a data center, and turning an overwhelming list of numbers into a clear visual story.

Drawing has also been about journey and discovery for me, often starting without clearly defined objectives and letting the process unfold and dictate the outcome. Not always drawing the reality that is infront of me but creating a new reality. Paul Klee described this process elegantly when he stated: "Art does not reproduce the visible; rather, it makes visible. " (Chipp, 182) It is this last aspect of drawing that I have always enjoyed the most. This mode of drawing allows me to find questions that I wouldn't otherwise think to ask. The process has always delivered me better ideas than simply brainstorming or even writing.

Throughout my life I have changed the tools I use for drawing, starting at a young age with markers, crayons, and pencil, eventually learning to paint with oils and watercolors. Painting is a process which I still consider drawing, only with a brush. In my undergraduate studies I learned to make linoleum prints and then intaglio prints (etchings, drypoint etc.). And then it happened... about eight years ago I started using the computer as a drawing tool. Now I use it almost exclusively. It wasn't until the last year of my graduate program that I asked myself, "Why?"

At first, I thought I used computers to draw because that was the only way I could make a career of doing

Torus Brush
Processing, 2009

what I have always loved most: drawing. There was a good argument for this, as I don't know anyone personally who lives off of drawing without using a computer (with the exception of a family friend who is employed full-time as a political cartoonist). There had to be more to this permanent switch in tools, so I asked myself what computers are good at, and came up with a list six key things.

I decided computers are good at illustrating space and time, facilitating interaction by managing input and output, storing and presenting data, running and storing complex algorithms, and distributing information to a large audience at little or no cost.

When I thought about this list relative to the history of drawing, I made a dramatic realization that these issues have long been of primary interest to artists using analog drawing tools. The computer was not just a fancy calculator for crunching numbers, but also a natural extension of the history of drawing.

For example, Marcel Duchamp's painting *Nude Descending a Staircase* explores several themes from my list. In this painting, Duchamp is playing with the Cubist algorithms for deconstruction of space, reducing the figure into simplified geometry and displaying it from several angles. He goes beyond space and tries to represent time by showing the figure in multiple positions on the same static 2D canvas, which also suggests motion. Unlike more representational paintings, Duchamp's painting required the user to interpret or decode the visual system of the painting in order see the motion of the figure and better understand the complex space it occupies.

Duchamp was also one of the first artists to draw attention to distribution in an increasingly mass-produced society. His ready-made sculptures like *Fountain*, an everyday urinal, challenged the viewer to question the value placed on art as craftsmanship and one-of-a kind artifacts by executing the conceptual act of placing a machined, mass-produced object in a gallery.

The more time I spent thinking back to art history and the history of my own drawing process, the more confident I became that the affordances of the computer were a match for the historical objectives of drawing. Using the computer, I am able to build and display models of 3D space on a 2D plane, generate drawings that change over time, turn data into drawings, write algorithms to generate near-infinite visual iterations of my work, distribute my work to a larger audience, and allow for greater interaction from that audience.

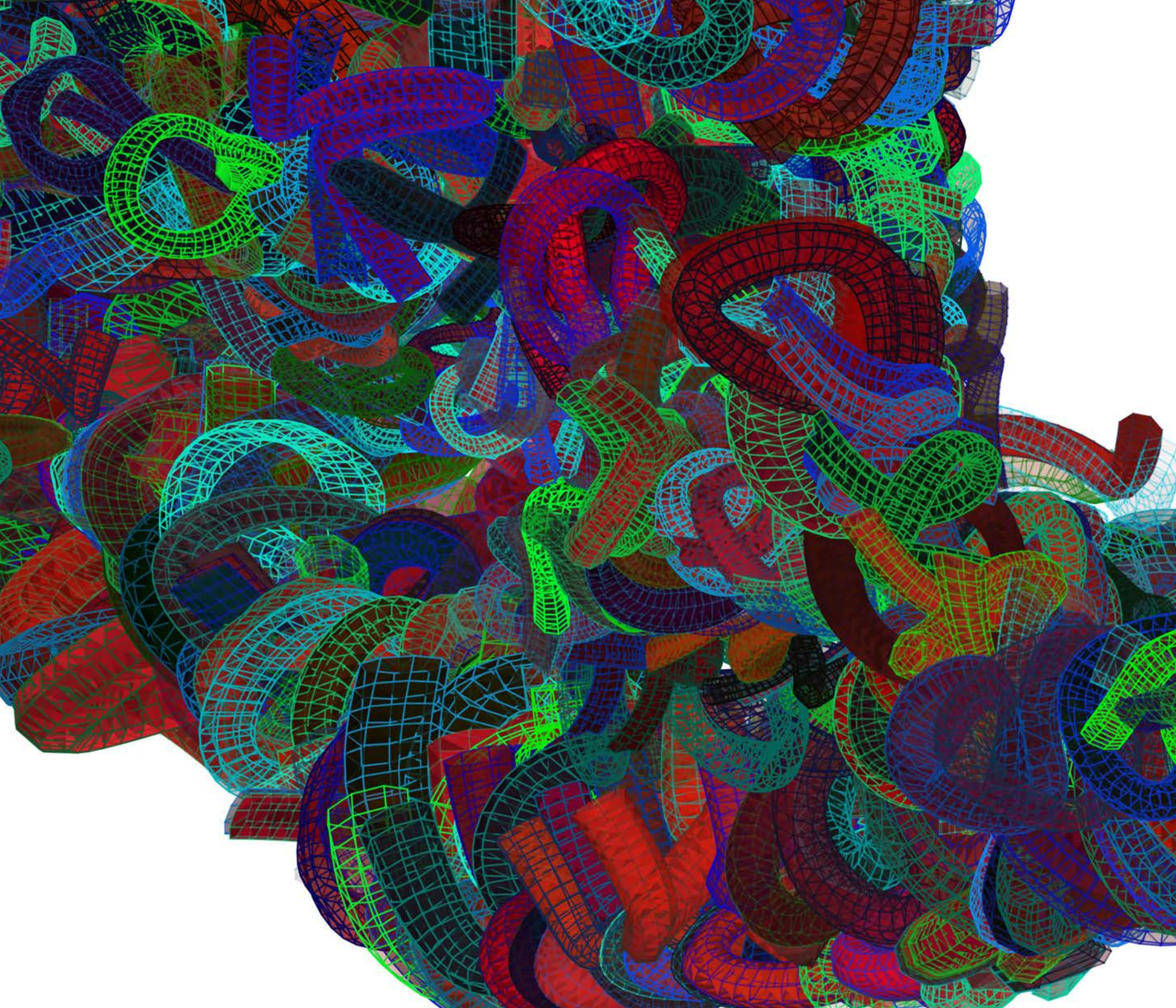
The computer also provides me a means for pushing drawing in new directions beyond its analog origins. When digitized, all media share a common language of

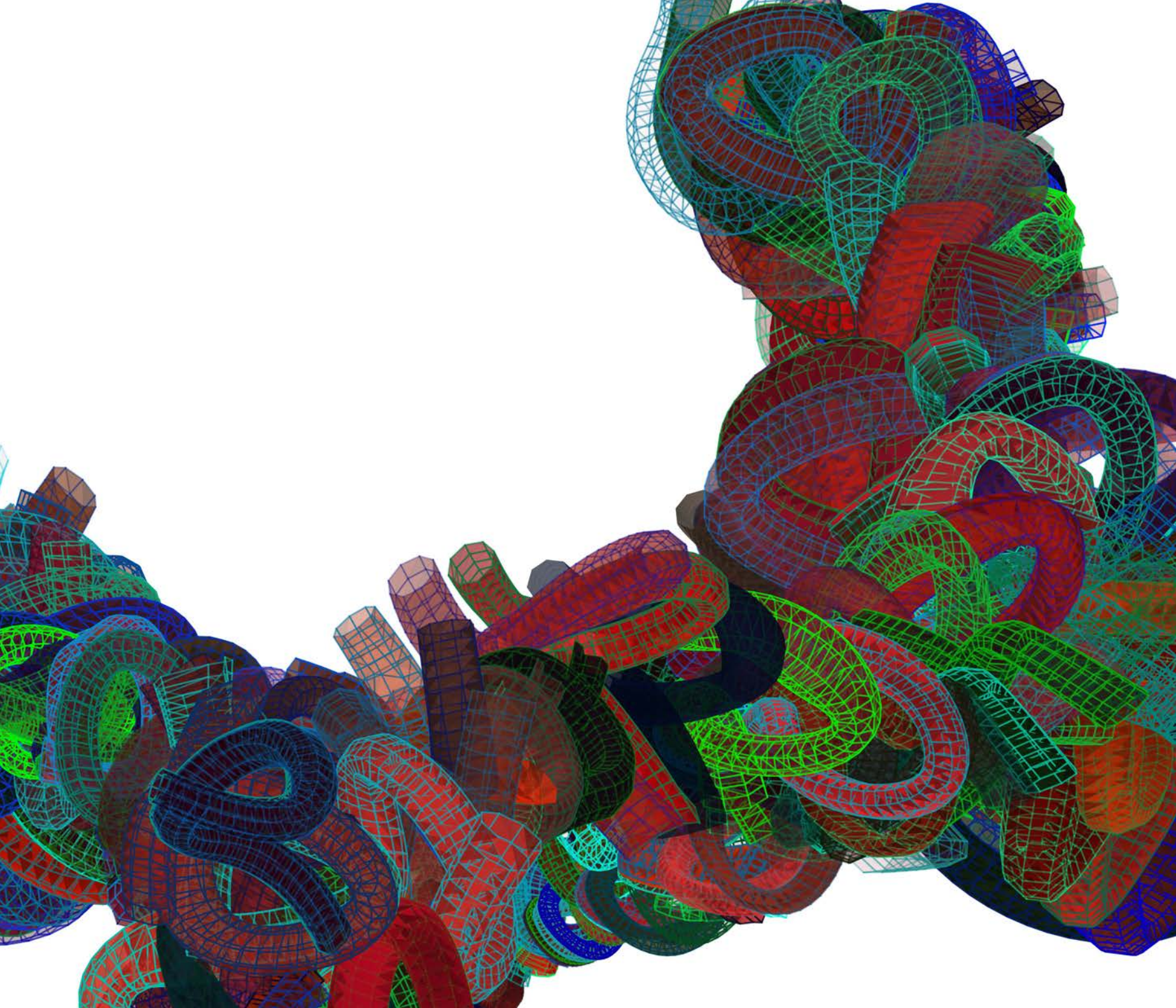
ones and zeros, allowing for uniquely digital, cross-pollination of media. For example, I can now make drawings with data taken from sound or compose music with data from video.

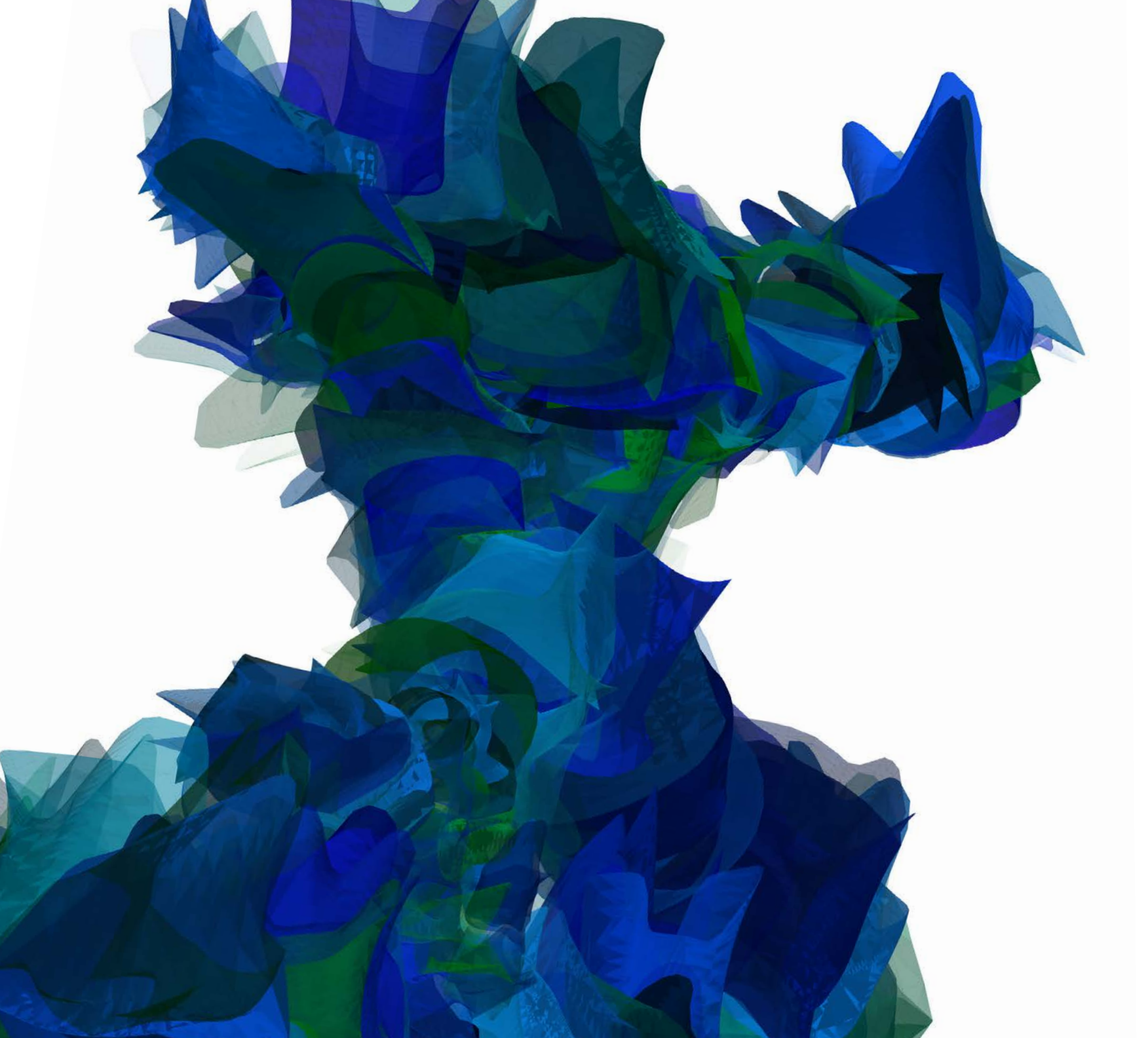
My thesis is comprised of a series of stories describing the process behind several of my drawing projects, both digital and analog. I have tried to capture the discoveries I made about the computer, how I use it as a drawing tool, and how it has changed the way I work as an artist.

What I learned from my projects is that it is best to involve people at every stage of development and that taking this approach is not only more effective, but more fun. This was counter to my instincts of going off and developing things in isolation based on theories of interactions. Sharing the process was more important than sharing the results. For this reason, I decided to write primarily about my process in my thesis. I wanted to tell stories, stories of successes and stories of failures, in the hope that others may learn from my experiences as I have from theirs.

Spread 3D Study #1
Processing, 2009







Space

I grew up Mormon and spent a lot of time drawing during the long Sunday services. I remember my dad being about as disinterested in the service as I was. The upside to this was that he would show me drawing tricks to keep me busy.

On one such occasion, he drew a square and then a second square diagonally offset from the first square. He then connected the corners and something magical happened. The two squares became a cube that seemed to jump off the page. Seeing how impressed I was, he followed the first trick by connecting two triangles and then two circles. Each time he made a new shape I was more impressed. For a kid who spent as much time drawing flat airplanes, flat horses, and flat ninjas as I did, this was a pretty amazing trick. Not only was it possible to describe 3D space on a 2D plane, it appeared to be incredibly simple using the tricks he showed me.

As with most people learning to draw, I spent a lot of time trying to figure out perspective and how to properly recreate the three-dimensional world around me on a two-dimensional plane. As an undergraduate student, I spent hundreds of hours drawing. This is when I realized that drawing complex shapes with proper perspective was a time-consuming and difficult task. Even drawing something as simple as an empty dorm room with a couple of shelves, some books, and a few bricks of Ramen noodles could take several hours to get right. Organic shapes like fruits or faces were even more difficult.

Learning to use 3D graphics programs on the computer restored the magic to the illusion of creating three-dimensional space on a two-dimensional plane. This is what I had seen when my father showed me how to make that first cube. With the computer I could simply push and pull surfaces in any direction I wanted, and the proper perspective would be calculated by the computer and applied to the scene. The idea that I can create a single model and show it from any angle without restarting the drawing process from scratch remains intriguing to me.

◀ *3D Brush Study #2*
Processing, 2009

3D Brush Tool

As the colorful and repeating sequence of rotating 3D shapes followed the cursor I was controlling with my mouse across my computer screen, I thought to myself, is this just more eye candy, or is there something here?

From over my shoulder, I heard my classmate Simon let out a coo of joy followed by a giggle. We had both signed up to take a summer course in the programming language Processing with guest lecturer Peter Kirn, a guru media programmer from New York City. Our second class was about to start. I had spent the first week exploring the 3D capabilities of the Processing programming language. Several classmates and professors told me that Processing had many advantages over Flash when it came to working with 3D, including access to the graphics card, which would allow for more complex shapes as well as a simplified way to generate basic shapes like spheres and cubes. Hearing Simon's positive response to the drawing program I wrote and seeing that he immediately wanted to try it reassured me that there was some potential for an interesting project here.

I had some light experience in programming languages and a background in 3D graphics, but had tried and failed several times to marry the two. Knowing that I would want more than just the simple boxes and spheres that Processing provided by default, I looked into methods for bringing more complex models Maya into Processing for further manipulation and interaction.

After finding a library of code to support the transfer of files from Maya to Processing, I began experimenting with building and animating the models first in Maya, and then manipulating them in Processing. Initially I created models of real-life objects, such as a table, lamp, etc. I wanted to make an interactive space, populated with realistic 3D models, similar to those in video games. Pretty quickly I realized that even Processing would have difficulty rendering lifelike objects with textures and lighting in real time. I decided instead to experiment with animating some abstract geometric shapes. I set up a processing sketch that would rotate my animated 3D meshes as the user dragged them around the screen. The resulting images looked pretty unique. It felt like a novel direction to move in.

I was excited about the 3D brush project, but equally excited about the next several projects in the Processing class, so I put the 3D brush project on the back burner.

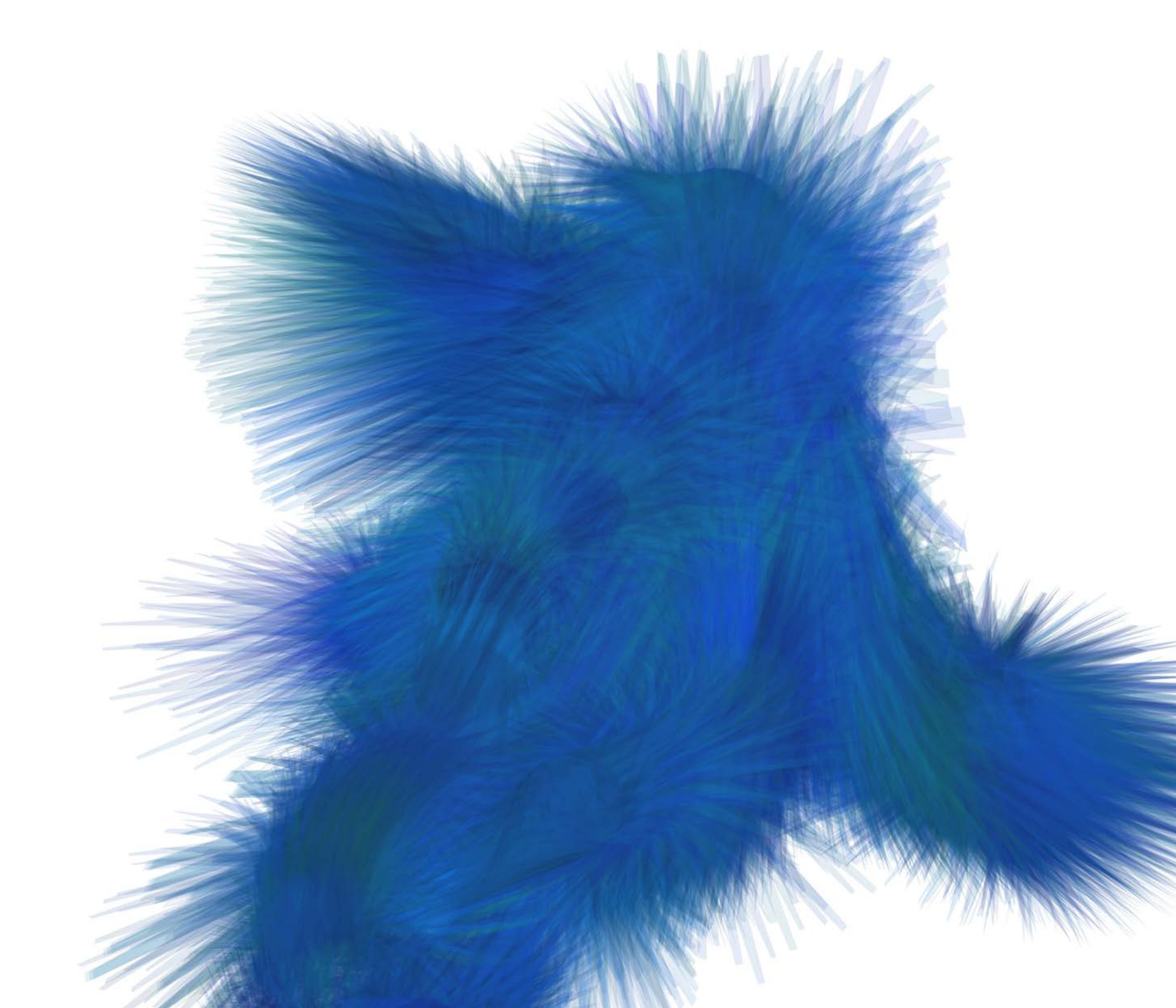
Knowing that I wanted revisit the 3D brush project, I signed up for an independent study in the fall of 2009. I had been looking at the algorithmic work of other designers such as the work of Jared Tarbell. Tarbell's

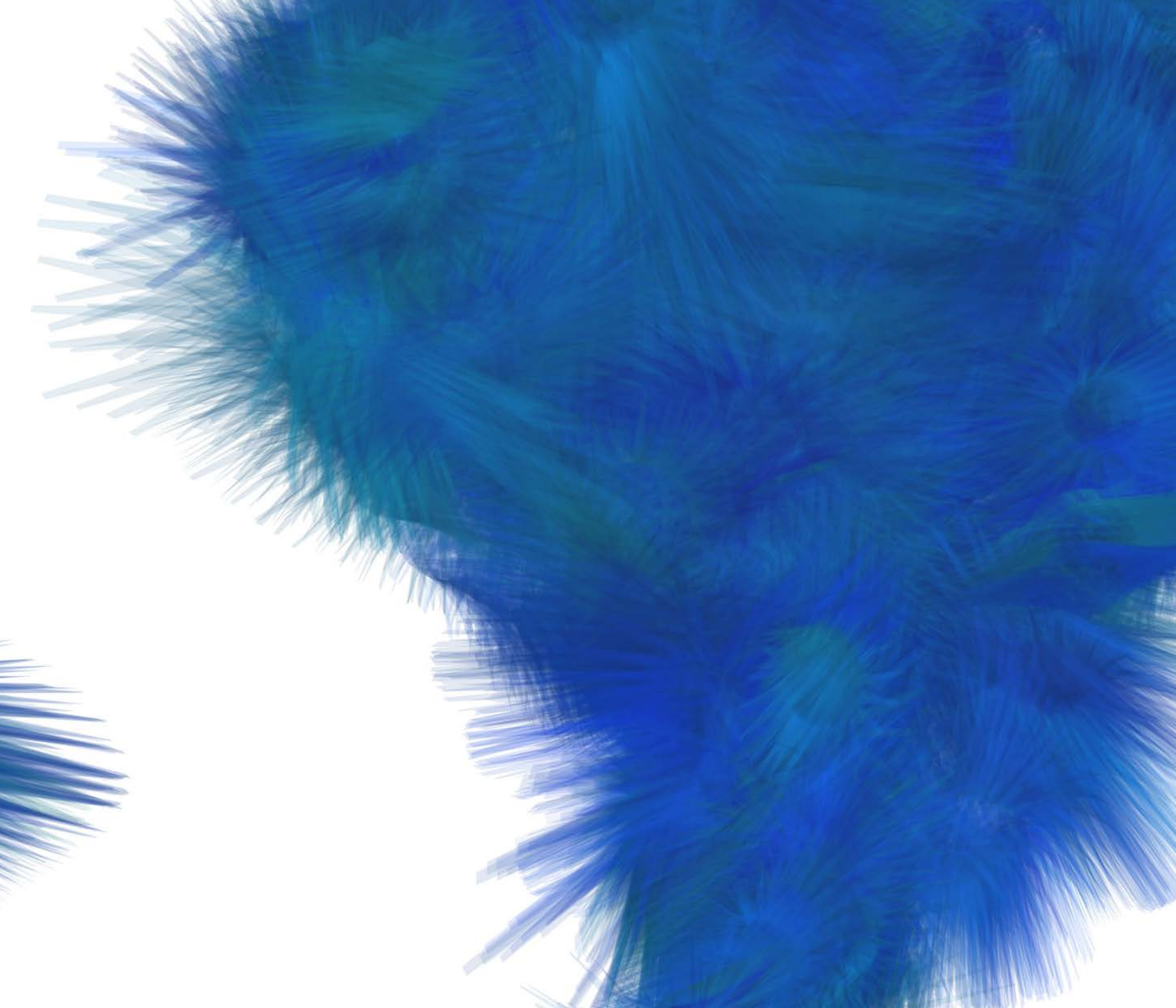
programming and design process resonated with my own process. In describing the creation of his algorithmic drawing, *Substrate*, he emphasizes the experimentation and discovery involved in programming drawings: "Before writing the program, I only had a vague idea of what it might look like. It wasn't until the first couple of bug-free executions that I realized something incredible was happening. The resulting form was much more complex than the originating algorithm. This particular quality of software is what keeps me interested." (Fry, 157) Tarbell's programmed sketches were intriguing to me for the painterly effects he achieved with them. Although generated entirely from code, Tarbell's images had a feeling of human gesture that was less present or not present at all in the work of other designers creating algorithm-driven artwork. I started thinking about designing a tool that would balance out the automated but organic results of Tarbell's algorithmic drawings with some degree of control from the user.

The first shape that I had used for my drawing tool was a torus or donut shape that appeared to dance as its mesh deformed over the span of 50 frames. The donut shape was set to repeat its animated dance in an endless loop and leave a trail of the previous frames in its wake. I spent several days experimenting with this shape, using wireframes, different lighting scenarios, color randomization, and increasing and decreasing the scale. I eventually settled on a scale that seemed appropriate for a drawing application, not so large that it immediately filled up the screen yet not so small that the details of the ever-evolving 3D mesh would be lost to the user. After becoming more comfortable with using the tool to draw, I learned how to balance out the idiosyncratic rhythm of the evolving shape with the control I did have over the motion of the brush.

It struck me during the development of this drawing program that my programming process and my drawing process were very similar. Casey Reas, co-creator of the Processing language, describes the simlarities of the two processes: "As with other types of art, software may be written through a process of intuitive exploration, and it can be written precisely to meet a goal." (Reas) I would often approach the code by making a small adjustment, sitting back and evaluating its impact on the overall image, and then going in and fine-tuning the code. As with drawing, I did not have complete control over the outcome, as I allowed for experimentation and surprises to influence both the code and the drawing.

Spreads 3D Studys #3, #4
Processing, 2009





Time

Taking Mike Golembewski's class in creative projects made the underpinnings of computer graphics seem less elusive.

Mike was serving as adjunct faculty in the spring semester of my second year. In the first two weeks of Mike's creative projects class we worked in groups. Our first assignment was to combine and rethink two objects which would be selected randomly from two lists. The first list was comprised of objects that function as containers, including drawer, back pack, closet etc. The second list included technological objects such as scanner, webcam, digital screen. My partner Elaine Froehlich and I were assigned backpack and webcam.

We had a great time brainstorming different ideas, and ultimately decided that the webcam would pull in live video from the front of the bag as the user walked around and display the video on the outside flap when the bag was placed on the floor, playing back the journey. We also thought filtering the video would make the displayed video more interesting, so we messed around with some webcam video in After Effects to simulate what that might look like.

Seeing that we were interested in filtering video, Mike set aside some class time to explain how digital video is drawn to the screen, and how real-time filters can be written in Processing to affect the display. I have worked with digital video editing software for about 10 years now, cutting and pasting clips, adding music and and titles, and occasionally adding filters for color correction or transitions between clips. Software like Adobe Premiere and Final Cut Pro has made it possible to edit video without ever understanding how it is pulled into the camera and drawn to the screen.

As Mike explained to us, video is drawn to the screen, many frames per second, pixel-by-pixel, left to right. It starts in the upper left corner of the screen, moves to the right corner, when it hits the right side of the screen, it then moves down a row, similar to the motion of a typewriter. Despite the speed at which this all happens, we can ask each pixel for information about its position on the screen, its color value, and its brightness.

◀ *Home Office*
Processing, 2009

More amazing was discovering that these values can be changed on the fly.

With this new information, video had become a much more interesting area to explore.

The next week in Mike's class, we were discussing how you can measure the difference between two different colors using a vector math function and the RGB values for each color. Although I still don't understand exactly how vector math works, I recognized this function as the same one I used to find the distance between XYZ points in my 3D programming projects. I had an epiphany that if I could use the same function to measure a series of three numbers for color values as I did for three points in space, and there might be some interesting visual mapping opportunities.

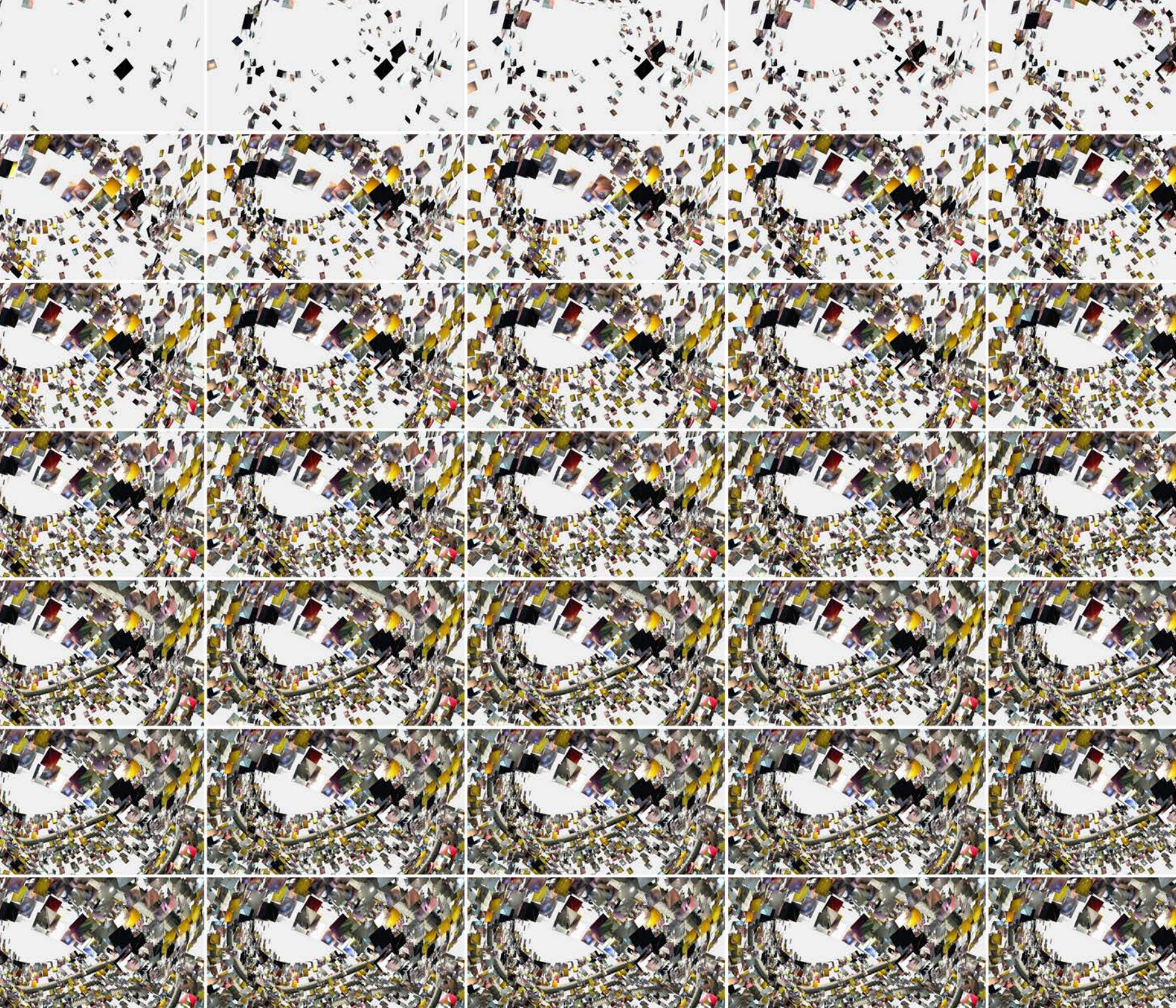
I wrote a program that would read every frame of a movie, get the average color of the pixels for that frame, and then correlate the R value to the X position, the G value to the Y position, and the B value to the Z position on the screen. Often the color for a frame would average out to some shade of gray and the frames from the video would cluster around the center of the screen. Every once in a while, a dominantly red frame would move off to the right, a green frame would move to the top of the screen, or a blue frame would move forward in Z space. I experimented with using the position of the brightest pixel in each frame to dictate the position of each frame as well, trying to get better separation. After experimenting with setting different thresholds for color and adding in some random motion, I had a pretty interesting program. As I put different color objects in front of the camera or pointed the camera at different light sources, a line of still frames would be drawn across the screen. My mapping wasn't perfect but this idea of drawing in space and time using video frames was turning up some interesting images and was fun to work on.

About a week later I decided to exhibit the partial project at the Doran Gallery at Massachusetts College

Gallery Sketch
Processing, 2009

of Art. Although not complete, I thought I could benefit from watching people interact with my project.

I'm not sure who he was, but one gentleman at the opening leaned over with curiosity and stared into the lens of my webcam, which lay resting on the white pedestal in the gallery. Clearly restraining himself from touching the camera (after all, it was in a gallery and it can be hard to tell what you are allowed to touch and what is off-limits), his eyeball was only about 3 inches away from the lens of the camera when he finally backed away and looked at the projection on the wall. There it was, the still image of his eyeball, projected as a plane in space, on the gallery wall, next to hundreds of other images taken from the camera during the previous 30 or so minutes. He went back to the camera, put his hand in front of it, stepped back again, and searched for the image of his hand on the wall, smiled, and then walked away.



Wave Sketch, Processing, 2009

◀ *Gallery Sequence*
Processing, 2009

I believe it was a motorcycle. I had been cutting up comic strips from the Sunday edition of *The Boston Globe* and blowing them up into large abstract shapes on the Xerox machine in the Whittemore Library at Framingham State College.

It was 1997. Scanners probably existed then, but I did not have access to one and would likely not have the technical skill required to operate one. The stylized engine and exhaust pipe of the cartoon motorcycle had these long elegant lines that I knew would work nicely in my collage.

I was working on one of many inspiring art projects assigned by professor John Anderson in my 2D design class. The assignment required us to use the photocopy machine to blow up the simplified, familiar images into several abstract compositions exploring balance of negative and positive space and variation of line weight. I had never done anything like this before, and developing a library of shapes that I could freely move around in combinations to create a new composition was really enjoyable. The forms that emerged were complex and interesting in ways that I would not have discovered if I were creating a preconceived abstract drawing. The element of chance and the speed with which I could experiment with new compositions by sifting through and juxtaposing images made this a memorable experience for me.

This project was memorable enough that over 10 years later it served as inspiration for a project at the Dynamic Media Institute. It was the process of creating these collages and not the end results that were so memorable and engaging for me. I decided to try and capture that process with a digital interface. This would require me to think about time and motion, two elements that were not present in the static drawings, paintings, and illustrations I had created in the past. I put together a proposal for a time-based, interactive collage tool:

"The tool would be based on the juxtaposition of simple 2D shapes to make more complex shapes. The interface would save the compositions created by the users into a database and play them back as inspiration for future users. When you approach the interface, the shapes

Collage Study #1
Flash, 2008

would move about the screen, occasionally assuming the compositions created by previous users. The current user could freeze the screen at any time and then rearrange the shapes, controlling position, scale, and rotation to form more complex imagery. Ideally, the results would express a meaningful and creative co-authorship between user and toolmaker."

I had been researching creativity and ways of facilitating it in users through the design of interface. One person who fascinated me in my research was Will Wright, the inventor of *Sim City*, *The Sims*, and *Spore*. At the time, I was trying to figure out if engaging people in the creative process was something that just happened or if there were design decisions that could be used to increase likelihood of success. Wright was of particular interest to me because he had developed wildly successful and engaging interfaces that allowed users to creatively combine existing elements already present in the game. I figured there had to be some type of formula for creative engagement. I was determined to figure it out.

In reading everything I could find about Wright, I found this quote from him: "What is the simplest possible system that I can build, that for you is going to decompress into the most elaborate set of possibilities?" (Moggridge, 381) This elegant statement made a lot of sense to me. I had been thinking that interfaces that facilitated creative experiences came from complexity and that complexity came from a wide variety of available actions and components to act upon. What I was missing was that complexity can stem from simplicity. I realized that I wanted to try and design a simple system that could unfold over time into a more complex series of interactions.

I thought back to the simple collage project that I had worked on in my 2D design class and how it had constraints for components and had simple rules, but also served as an engaging system for creative process. I decided to use it as the inspiration for a digital collage interface. I needed to find some interesting shapes that people could use in combination to create more interesting shapes. I decided to vectorize several artworks from 20th Century artists including Jean Dubuffet, Jean Arp, and the collage work of Henri Matise, and break them apart to provide a library of shapes. These artists appealed to me for this project because their work was comprised of simple, graphic, hard-edged forms.

I knew I wanted people to be able to combine these forms, rotate them freely, and change the scale, just as I had in my Xerox collage project, only this time with a computer. I was worried that this experience may not be as fun in a digital environment, so before I started coding up the interface for scaling and rotating the shapes, I played with the shapes in Illustrator, making several

◀ *Collage Study #2*
Flash, 2008

compositions. Convinced that this was worth exploring, I moved forward with my programming.

While experimenting with Illustrator, I realized the steps required to move, rotate, and scale the shapes were actually quite complex. Brainstorming ways to reduce this complexity, I came up with an idea for a single-click interface that would cycle through functionality but use the same gestures. I centralized the code so that each shape was an object with the same behavior; I could have one shape or a thousand shapes using the same amount of code. When you clicked on the shape, it went into the first stage of interaction and you could drag it around the screen, juxtaposing it with other images. A faster gesture with the mouse would throw the shape across the screen, bouncing off the edges until it decelerated to a complete stop. A second click allowed the user to rotate the shape by moving the mouse to the left or to the right; changing the scale was accomplished by moving the mouse up or down.

After many months of programming, I had an interface that I was quite happy with. I had solved several technical problems, devised a novel method for scaling and rotating shapes, and the motion felt quite organic. However, I was reluctant to have anyone use my interface. What if all of this work resulted in a dud? I eventually brought a variation of the interface into class and had several classmates try it out. While they had fun with it, none of them actually created the collage-type images I had in mind. I had worked in isolation too long and made too many assumptions. Developing a single formula for facilitating creative engagement was starting to feel like a pretty weak idea, but it didn't stop me from further researching this idea.

Studying Will Wright led me into more research on complexity derived from simple systems. Wright had mentioned several times that he was inspired by *Conway's Game of Life* and cellular automata, so I decided to investigate this further. Cellular automata are "grid-based systems that vaguely resemble game boards." The squares or cells can become occupied or vacant based on a simple set of rules which often lead to complex, emergent behavior. *Conway's Game of life*, created in 1970 by mathematician John Conway, is probably the best-known example of cellular automata. His rules dictated that an occupied cell would be "alive" and vacant cells would be "dead." When set into motion, each cell would change based over time on the state of its neighbors. The amazing part about these simple systems is the variety and complexity of patterns that emerge. (Salen, 161)

I was staring to understand why Conway was so inspirational to Will Wright in creating his games. Because Wright used only simple rules and few components,

his games were easy to learn, but his understanding of systems meant the games had an extremely diverse set of outcomes. This again started to sound like a formula that could be applied to a tool for image-making, whether it be drawing, collage, or building.

I became more interested in systems and simulations as a means of creative engagement. A classmate suggested I pick up a copy of Mitchel Resnick's book *Turtles, Termites & Traffic Jams*. In his book, Resnick describes the development of Starlogo, a program Resnick wrote for students to model and simulate the behavior of decentralized systems like traffic jams. Before StarLogo, Resnick had been working with students to learn about decentralized systems using one of his other projects, Lego/Logo, and realized it was not the ideal environment. He explains: "To explore phenomena like these, I decided to go back to the computer screen, to work with virtual creatures in virtual worlds. Virtual worlds offer many advantages. In virtual worlds it is easy to create large numbers of creatures. It is easy to give new sensory capabilities to the creatures. And it is easy to set up and control precise experimental conditions." (Resnick, 31) Like Will Wright's *Sim City*, Resnick's StarLogo interested me for the variety of potential outcomes the user could arrive at while manipulating only a small number of components.

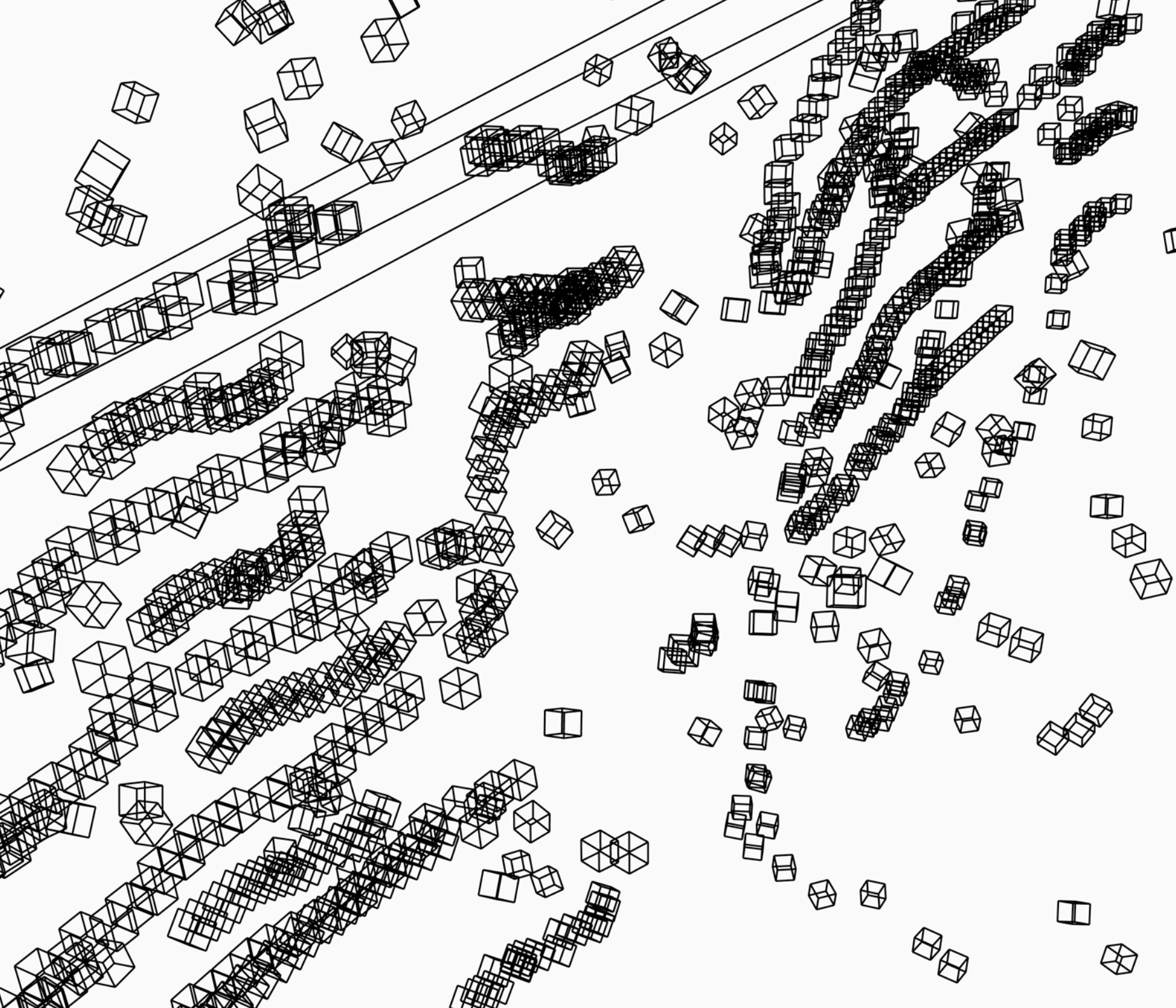
At this stage in my studies, I was doing more reading than making. After reading *Turtles, Termites & Traffic Jams*, I was hungry for more reading on systems. In browsing the bookshelves at the local book store I came across Noah Waldrip-Fruin's book *Expressive Processing*. In the introduction he states: "...computational processes are an increasingly significant means of expression for authors. Rather than define the sequence of the words for a book or images for a film, today's authors are increasingly defining rules for system behavior." (Fruin, 3) I was now more convinced than ever that all the important people were creating systems, authoring experiences through the development of rules.

It was the beginning of my last summer at DMI, and I started feeling the pressure to come up with a thesis project. Inspired by everything I had been learning about simulations and systems, I started a second attempt at creating an interface that would allow the user to set simple components into motion with the hope that these components would evolve into interesting compositions over time. I had a decent understanding of the Processing programming language and I was impressed with its 3D capabilities. Inspired by the simplicity of the Lego building system and my research on systems and cellular automata, I set out to design a simpler version of the collage project in a 3D environment.

This was an unusual approach for me, to prove out a project idea based on reading and having a detailed plan of what the project would be before starting to make it. After several weeks of programming, reading tutorials and "borrowing" code, I had an empty environment full of gray cubes that would highlight in red when selected. Once selected, you could move the cubes around the screen and combine together to make more complex shapes. After you built up a model with the simple cubes, you could apply simulated physics to make the model fall apart. This was my attempt at further applying the idea of simplicity leading to complexity as a means of generating an interface that could facilitate creativity in the user.

I had my friends and classmates try the interface. The general response was that it was impressive from a programming standpoint but the glitches it did have prevented people from wanting to build and explore with the interface for very long. While this was a major success for me from a technical perspective and a vast improvement on past projects, it was still missing something. It was not engaging the user. I also felt like working with such a predetermined outcome was unnatural for me. It felt more forced and less enjoyable, not as open to chance and discovery. I was trying too hard to force my process to mirror the research I had done on simulation and systems.

Cube Builder
Processing, 2009



Interaction

"Art teachers are the biggest bullshit artists on the planet!" Oh, my God, I thought... what have I done? It was my first day as a high school student teacher.

I was hardly a year or two older than most of the high school students in the classroom, and the student who just called me a wannabe "bullshit artist" looked like he was a year or two older than me and wanted to beat my face in. I forget his name, but he had the same stupid glower of the seemingly infinite number of jocks that had made my own experience as a high school student so miserable.

I had voluntarily gone back to hell. Somewhere in my head I thought that being a high school art teacher would give me the opportunity to help struggling students find purpose and shelter, like my art teacher had done for me. I had forgotten that art is a requirement for all high school students to graduate and the majority of those students would rather be playing kickball or doing just about anything other than sitting in art class. I don't mean to imply that they weren't good kids; it's just that not everybody is that into the whole art thing.

Some of the students clearly enjoyed being there and enjoyed the projects they were working on. There was the kid with bushy black hair riddled with dandruff drawing an army of robots in compromising positions. The girl with no eyelashes and cigarette burns on her arms drawing a beautiful still-life with colored pencils and cray-pas. And the painfully stereotypical teacher's pet: a bright, smiling Chinese girl with a Dartmouth College sweatshirt. She had single-handedly resuscitated the ceramic kiln from near extinction and proudly displayed a long row of elegantly crafted organic vessels. Two years earlier, I would have been hanging out with these three kids after school, perhaps even starting a crappy garage band with them. They were my people.

For the majority of the semester I just watched and took notes as Mrs. B. directed her classroom. She was a great teacher, but I couldn't help but notice how all the students worked in isolation. Their projects seemed too

◀ *Group Drawing #1*
Marker, 2009

precious to them. They were afraid to try new things for fear of failure and judgment. I remembered these observations when Mrs. B. gave me an opportunity to create an assignment of my own, and thus, full control over the class.

There was a graffiti problem at the school, and due to this, one side of the building had been sandblasted several times that semester. I remember thinking about how the anonymity of that act of graffiti must have been freeing for a high school student under constant surveillance and scrutiny. Inspired by these anonymous acts of graffiti and a desire to get the students to collaborate, I decided that my assignment would involve each student drawing on a large piece of paper for one minute and then passing it to the person on their right. They would do this until they received the drawing they had initiated.

I brought in my cheap dorm room CD player so I could play inspiring music for them while they worked. Before playing a selection from some crazy experimental sound band from the '90s (Praxis, I think), I laid down the ground rules of the assignment. I told them the only rules were to draw for that one minute and then pass to the right. I didn't care what they drew. I told them I would not censor them. "Serious? We can draw anything?" they asked before letting out a collective giggle. "Anything you want," I said.

I started the music, set a timer and said "go." Everyone started drawing with an urgency. Some drew hearts and smiley faces, others drew abstract squiggles, some chose to write words. It was awesome. One student decided to draw the same image of two stick figures having sex over and over again. Another student further down the line was offended by the X-rated stick figures and would turn them into flowers, rainbows, or just cross them out. At the end there were 30 unique compositions created by the class as a whole. We were able to critique them openly, because there was no clear single author. Mrs. B. hung them side by side like a giant graffiti wall in the hall outside the art room. She told me she would definitely be doing that assignment again and even asked for the name of the CD I had played.

I have re-purposed this drawing exercise several times since my student teaching days, most recently in response to an assignment on anticipation at DMI and again as a visiting lecturer at Framingham State College (my alma mater). Turning drawing into an exercise in socialization and interaction, instead of isolation, appeals to me. It puts more emphasis on process, the part I value most, and less on the quality of the artifact or finished piece.

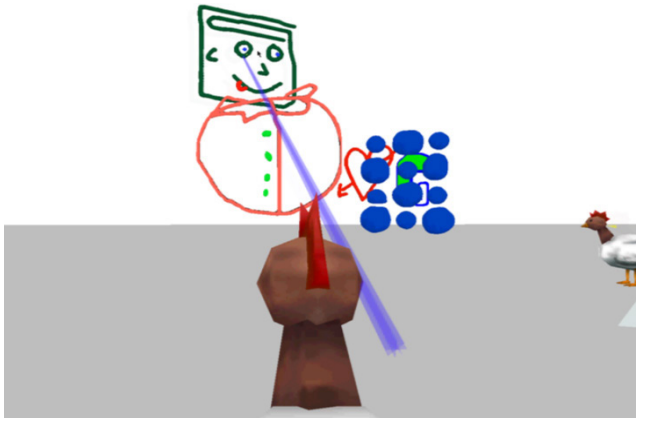
Group Drawing #3 ▸
Processing, 2009

Entering into my final year at DMI, I realized I had worked on several projects that I intended to be platforms for collaboration, but always hesitated to bring them to the level of user testing for fear that they would fall flat.

Initially I thought an interface with a blank screen that allowed users to collaborate in 3D was the way to go. I had tried *Second Life* and several other spinoffs and found them to be too cluttered for serious collaboration. In my second semester at DMI, I decided to try and create a minimalist collaborative environment where people could create models and make drawings together in three dimensions. The environment would combine 3D software like Autodesk Maya and Google Sketchup, with the networking capabilities and social interaction of *Second Life*. It would be a blank canvas without artificial terrain and over-the-top avatars, or so I hoped.

Knowing my limitations in programming, I sought out a platform to build upon that would already have resolved the 3D elements such as cameras, texture mapping, and navigation. I eventually settled on Croquet, a program that Alan Kay developed along with several other computer graphics luminaries. According to the Croquet consortium website, Croquet is "a powerful open source software technology that, in the form of the Croquet Software Developer's Kit (Croquet SDK), can be used by experienced software developers to create and deploy deeply collaborative multiuser online virtual world applications on and across multiple operating systems and devices." (Croquet)

I must have glossed over the part about "used by experienced programmers" because I spent the better part of a semester trying to resolve technical challenges to get a prototype of my empty 3D world up and running. At the end of the semester, I presented a blank screen with a simple grid on which several people could log in as 3D chicken avatars and draw in 3D space. I planned on replacing the chicken avatars with floating 2D planes mapped with live webcam video, but was not able to make the changes in time for the presentation.



Collaborative Chickens
Croquet, 2008

The presentation was a flop. The silly looking chickens on the minimalist background combined with the awkward drawing tools, and the inability to demo live without crashing, made it difficult to express my intentions. Because getting the platform up and running that semester used up the bulk of my time, the majority of the project relied on existing tools and functionality already built into Croquet. It took another two semesters before my classmates stopped picking on me for my chicken-drawing collaborative interface.

Although I had been thinking of this collaborative 3D environment as a potential thesis project, I abandoned it as there were too many technical challenges and I had sensed little to no interest from people in using such a tool.

I decided that I needed to do more research on collaboration and creativity in order to make my future projects more successful. I read book after book about systems, game design, interaction design, design patterns, and the philosophy of creativity in the hope that they would lead me to a set of best practices for designing interfaces that would facilitate creative interaction. By this point in my thesis development, I had realized that I wanted to focus on drawing as an interface for creativity, but I was also realizing that my approach of researching in books for a set of rules for designing creative interactive interfaces was becoming counterproductive.

Luckily, in the start of my final year at school, I realized what was missing from my previous collaborative projects: collaboration. I had come up with these projects in isolation and worked by myself at the computer for long hours with a predetermined vision of a finished piece and lots of false assumptions about how people would interact with my projects.

It was around this time that I had a conversation with Dennis Ludvino, a classmate of mine who was

working on a collaborative writing environment that had a lot of crossover with my interests in creating a collaborative drawing environment. I approached him about "collaborating" on making a shared canvas together. He thought it was a great idea, so we started working out the details. It turned out to be one of the best decisions I made while at DMI.

We started out by using a pre-built Processing sketch that would allow two people to share a screen through a network. We immediately ran into limitations. The software would crash, it only supported two users, and we needed to adjust the code every time our computer's IP address changed. These types of technical problems usually pop up in the beginning of my projects, but it was already less intimidating having someone else to work on the problems with.

With zero networking experience between the two of us, we went off and researched the different problems, checking in periodically with different solutions that could potentially advance our progress. By sharing the work, we were able to make several breakthroughs on the technical side in a relatively short amount of time. The first breakthrough was learning about and implementing Dynamic DNS on the server side of our application. This allowed us to use my home computer as the server and not worry about if and when the IP address was going to change, which would break our network. The Dynamic DNS would check for this change and automatically update without requiring us to change our code.

The breakthrough we are most proud of was finding a way to recognize multiple users. We had a theory that if we were already passing the X and Y positions of our mouse back and forth, why not pass an additional identification number to let the other users know who the information came from? Then we could write code to recognize each user individually as the information was sent back and forth and add many more users. This may sound simple, but having come up with this solution completely on our own, with such little programming experience, left us thinking there was very little chance that it would actually work. It was extremely gratifying to see our homegrown solution expand the demo code we had started with into a truly multiuser interface.

After solving the main technical issues, the collaboration only got stronger. We felt like Alexander Graham Bell and Thomas Watson when we were finally able to get the shared canvas up and running with Dennis drawing from his home in Lowell, Massachusetts and I, many miles away, drawing from my home in Ashland Massachusetts. The first drawing came together rather quickly. One of us, I don't remember who, started drawing a

◀ *Unicorn & Rainbow*
Processing, 2009

horse, the other added a single horn, turning it into a unicorn, and then we both collaborated on a rainbow. The drawing was hideous, but the experience was fantastic.

At some point early on, we made a mistake in the code, and there was a line being drawn between the two users, in addition to the lines created by the users mouse. This changed the drawing process quite a bit as it filled up the screen at a much faster rate. It also created some very intriguing shapes. We were pretty sure we understood what was going on but we invited Dennis's fiancé Krista and my wife Erin to join us, creating a four-person drawing to confirm our assumptions. We all had a great time making strange abstract webs of network lines. After we made the drawing, we combined all four drawings, one from each computer, into Photoshop layers where we could study the relationship between them.

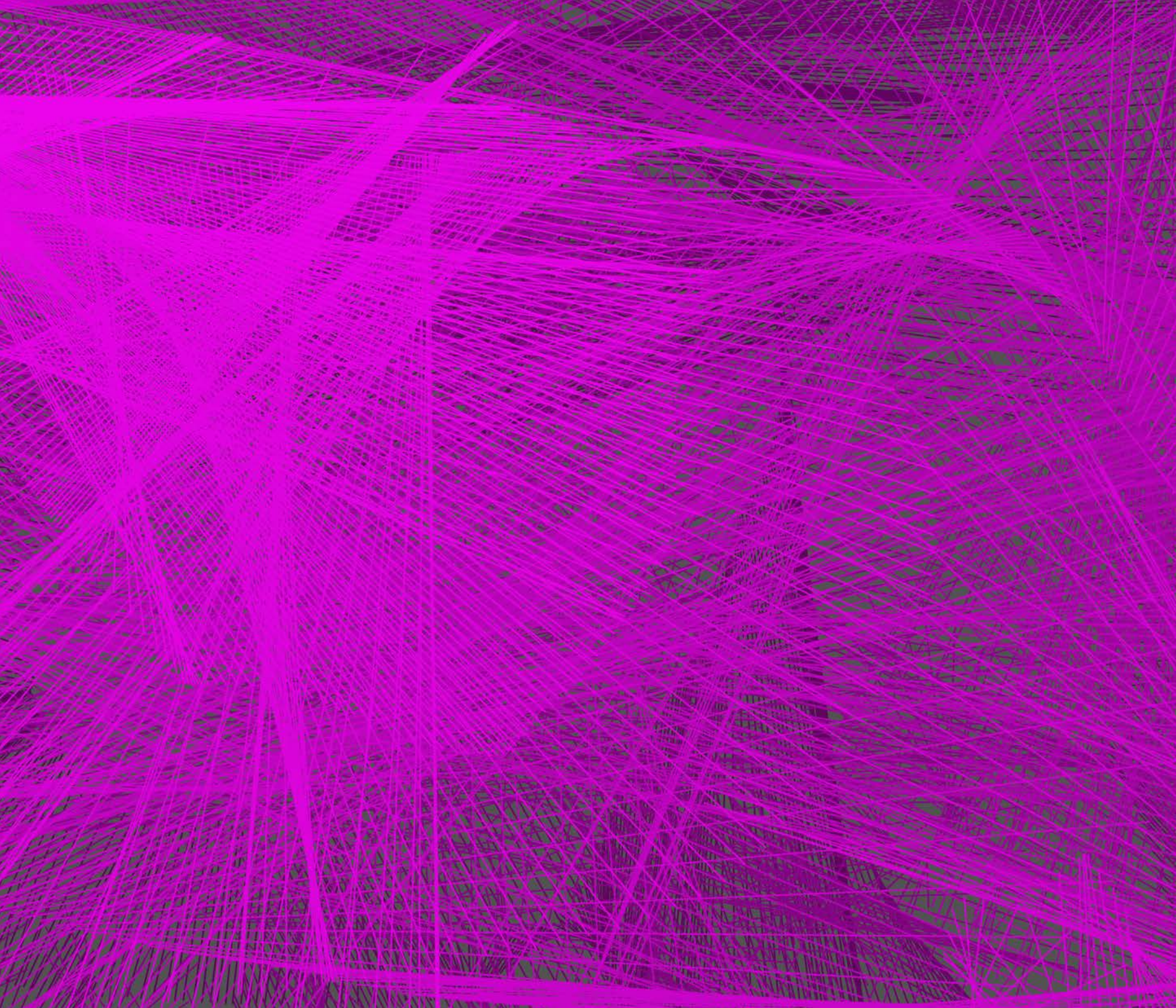
Working with Dennis kept this project moving in a fun direction. We both felt like there was something really cool about drawing at the same time as someone else, in different locations, on the same canvas. The more people we let use it, the more confident we became that we did not need to justify the purpose of the interface. It was fun and engaging and every time we came together to work on it we discovered something new.

The drawings we created using the program were truly collaborative: often juvenile in their subject matter (robots, unicorns, middle fingers), but always in response to one another. Typically, someone would draw a circle or a box, to which the other users would add a face or body. Psychologist Manfredo Masseroni devised a similar exercise where by the reader would create a simple scribble and transform it into a bird by simply adding a small circle O and beak < shape. Colin Ware describes how the "deliberately 'random' scribble is often used by artists and designers as a way of liberating their creative process from stereotyped visual thinking. The perceptual mechanisms are exquisitely tuned to find meaning often from slender evidence, and this is why meaning is so easily found in meaningless scribles." (Ware, 154)

Our meaningless scribbles would transform into a variety of characters, sometimes appearing as Santa Claus, an alien, or a portrait of one of the users. Inevitably, someone would start to create a landscape for these characters to live in, drawing a simple line for the ground, stylized trees, or mountains in the distance.

As the screen filled up with lines and became more chaotic, the subject matter would also shift toward chaos. Someone would draw weapons for the characters, guns or axes attacking the other characters on the screen. This aggression would spill over from the battle of the characters on the screen to personal attacks between users. These personal attacks were in the form of someone writ-

Network Lines
Processing, 2009




SPACE
BUS


Saturn

ing a comment about someone else's mother, or a series of drawings of hands giving the middle finger.

In our last semester, we decided that we would continue to work on our collaborative drawing project together, but this time we would bring in even more people to help. Instead of thinking of the people we shared our project with as user testers, we thought of them as co-authors for our project. We would use the project with them, and implement the feedback they gave us on the fly.

For example, we created a series of drawings over the network with Scott Murray, a classmate living in San Francisco. As we made our drawings, Scott mentioned that the canvas seemed to fill up too rapidly, making it hard to continue the drawing. He suggested we have the drawing fade over time and the three of us implemented the code on the spot.

In another session with once classmate and now teacher, Colin Owens, we discussed ways to decrease the delay between user gesture and the image appearing on the screen. Colin wrote a version of the code that helped us speed up the lag making the program closer to real-time. We used Colin's code in combination with some code we worked on with my younger brother Matt, a software engineer by trade, to push the code into the server side, further reducing delays and allowing for additional users without having to adjust the code, improving the user experience.

Instead of fearing the feedback of the users, we viewed them as assets to help improve our project. We discovered that most people were excited to be included in the making process, instead of just the using process, and the more people we included, the more fun we all had.

Working collaboratively turned a stressful semester of trying to finish my thesis into an enjoyable time of collaboration between friends, classmates and family.

I am not sure if we will continue work on this project after graduation, but I am positive that the experience has changed the way I will work in the future. Working in isolation, guessing at how people will interact, and performing staged user testing was not really working for me. Getting together with other creative people and working through ideas together from the beginning is a lot more fun and a much better approach for designing interactive projects.

◀ *Super Chute*
Processing, 2009

Data

A week before my first thesis meeting with my advisor, I went on an annual getaway with my father and brothers we call the "Bailey boy weekend" camping trip. It was the last weekend before I started my final year at DMI.

After three days of drinking beer and tequila, and incorrectly playing a partial copy of the board game Risk, my brothers, my father, and I sat around the fire to continue the 31-year-plus-long conversation about software, hardware, and technology. As the least technical member of the group, my role in these conversations has generally been as a listener.

Strangely, the conversation shifted to a debate about analog vs. digital, which is something we frequently debate at DMI. But to hear three engineers talk about it made it seem like a totally different conversation. My father played the role of the defender of all things analog, saying that, "There is no such thing as a unique digital concept." My dad claimed anything digital was created in analog format first, and when you convert something from analog to digital, it's inevitable that you lose information in the process.

My brothers Matt and Brian, who argue for sport, had a rare surrender, agreeing with this concept of there being no unique digital experience and analog's superiority. While they were briefly paralyzed by the unfamiliarity of consensus, I saw a rare window of opportunity to say something without being talked over or shut down. I disagreed with their outcome, and started explaining why, but I was unable to get the ideas in my head to come out of my mouth as words that made any sense. Damn you, Jose Cuervo!

Despite the tequila haze that followed me for the next several days, I had locked onto this concept of a unique digital experience. I thought about the projects I had made at DMI. Why were they digital? Just because I was in a digital program at school? More directly, what was I doing with a computer that I couldn't do utilizing analog tools and why?

I eventually had one of those "eureka" moments: most traditional analog media have entirely different inputs and outputs. For example, a saxophone player blows air across a wooden reed (input); out comes the

◀ *Giant Steps*
Processing, 2009

sound of vibrating air, which we call music (output). This is dramatically different from a film camera in which film is exposed to light and treated with chemicals (input) to create a sequence of images that can be projected in succession (output). Equally different from the two previous examples are drawing and painting, which both employ pigment and a stylus (input) transferred to what is usually a flat surface as a composed static image (output).

I decided one of the unique opportunities of digital media is that all traditional media, which had entirely different inputs and outputs, now share a stage of development where they are all data. The result is not a crude approximation of an analog experience but an entirely new experience which is uniquely digital.

Drawing With Data

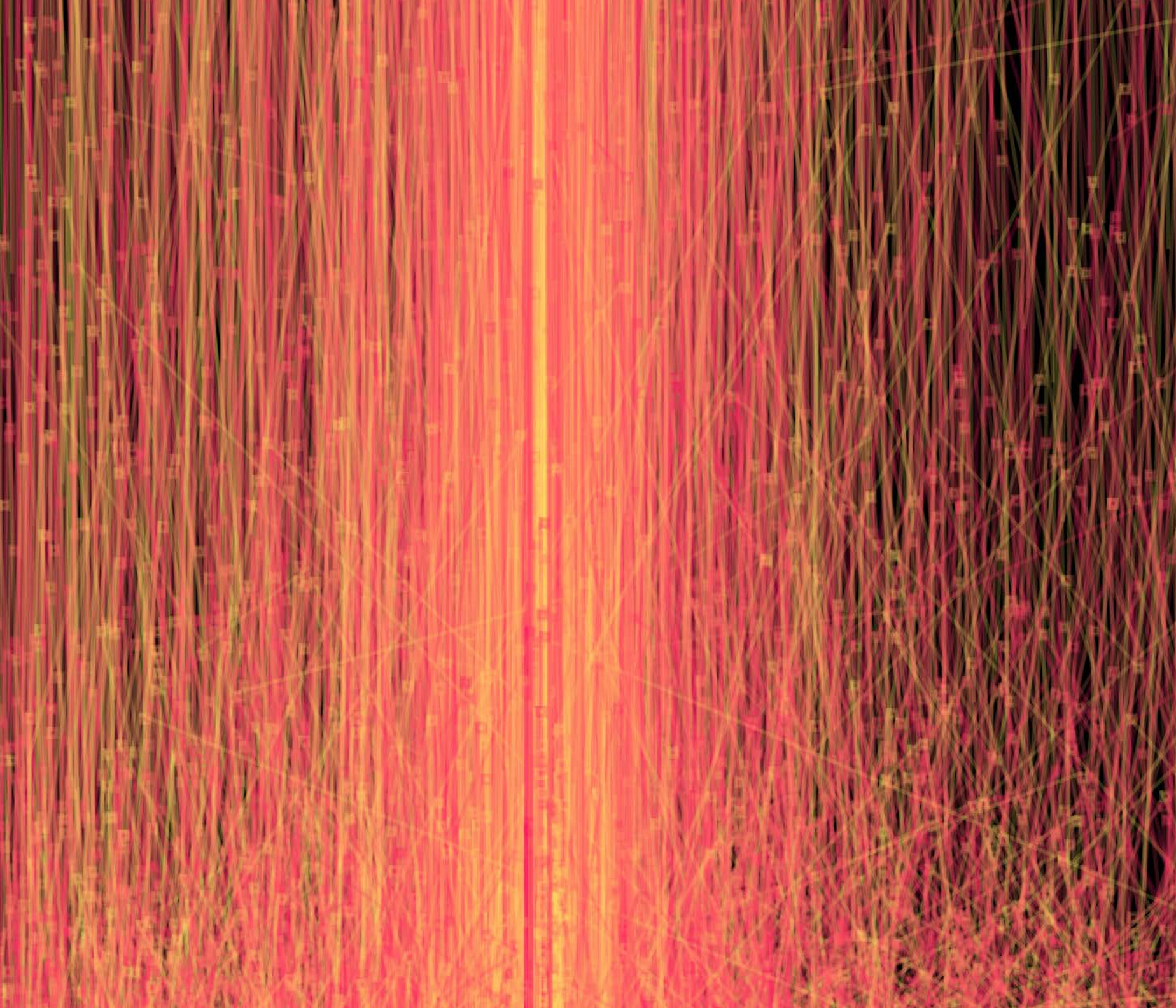
After having the epiphany about reducing all media to a common language of ones and zeros, or "data," I thought back to past projects I had worked on that took advantage of this. One project I created would draw lines in space using frames from live video; another project I created that summer used object tracking algorithms to create musical compositions from video data taken of different play dough shapes. The user could change the shapes and adjust the sound, or move the shapes left and right to affect the panning. In both of these cases I was driving one media using the input from another. In the first case video input was translated into a 3D line drawing output. In the second case the sculptural form of the Play-dough was recognized and translated through video as an input and then output as sound.

While these projects were already taking advantage of the cross-pollination that is made possible by digital media, I wanted to make several smaller studies with this idea in mind. At the time, I had also been thinking about the role of gesture in drawing. In my programmatic drawings, physical gesture became irrelevant. The way I pressed the keys on the keyboard did not affect the resulting drawing. I started thinking about the mouse gesture, and how I had thought of it more as a limitation for drawing than anything else. I eventually asked myself, Why I was writing off the mouse gesture as irrelevant? After all, I spend more time gesturing with my mouse than I do making just about any other type of gesture; it's the nature of working with computers. What if I captured the data from the everyday gestures of my mouse and turned them into a drawing? What would it look like?

Normally, I would spend a lot of time trying to write a program to capture my mouse position over time, but I knew I wanted this to be a short, experimental project. So instead, I found some free software that would track the X and Y coordinates of my mouse and save them to a text file. I turned on the mouse tracker and generated thousands of lines of data as I went about my regular computer related tasks. I tried not to change my behavior as I was interested in capturing the unmodified gestures of my daily mouse-driven activities. I realized pretty quickly that I would have more data than I new what to do with, so I grabbed a five-minute chunk of the data and put it into a format I could use to import it into Processing. Once I was able to get the mouse data into Processing, I started playing around with drawing different colored shapes and lines to represent the data.

At first, I thought of this exercise as a data visualization project. I tried to make it clear where the mouse started and stopped by adding circles at these departure

Mouse Map Drawing
Processing, 2009



points. I lost interest in that approach and started to tweak variables, and add new code to add randomization to the color and location of the lines, testing the drawing each time I made a change. The drawing transformed very quickly from a visualization of my mouse gestures to a more abstract and appealing image.

The dense network of colorful lines spanning the screen created a complex dimensional space. Often my goal in creating drawings is not to recreate what I see, but rather to create an entirely new visual space comparable in intricacy to the world around me, less familiar yet still inviting.

Initially, it seemed ironic that breaking from the data acquired from actual gestures had made the composition more gestural in its visual qualities. Unlike drawing with a pencil or brush, my programmed drawings relied more on conceptual decisions than physical gestures. My analog drawings required a learned physical dexterity and facility developed over time. That process was different yet strangely similar to drawing with programming.

My analog drawing process, particularly when creating more abstract work, involves making a few marks, stepping back, and evaluating. Often I spend more time evaluating than making marks. After that period of contemplation, I usually feel like I know where that next mark belongs.

Frequently when drawing, something outside of my control will happen, an accidental drip of paint, a smudge of charcoal, etc. These elements have become a welcome part of the creative process for me and frequently take the composition in new and interesting directions. When programming drawings, I have a very similar process. I start playing with code, checking in periodically to see what the resulting image looks like. More often than not, I am surprised by what comes on the screen when I run the program. Sometimes I like what I see, sometimes I don't. One of the great advantages of programming drawings is that my accidents are repeatable. If I tweak the code and like what I see, I can reuse that code in new combinations and turn what was initially an accident into a tool.

After completing the mouse drawing series, I realized I had a tool that could make drawings out of any data with a similar two-column format by mapping the data to the X and Y coordinates and creating lines. Thinking back to my ideas on driving one media with another, I started thinking about using this same code to make drawings out of songs. I knew that the music I played on my computer was really just a sequence of numbers at a base level, regardless of whether it was an .mp3 file, a .wav file or any other audio format. I experimented with different formats trying to find one that would produce

◀ *Harmonica Visualization #1*
Processing, 2009

data that I could use with my mouse position drawing program. It turned out that exporting music to the ASCII format produced exactly the type of data that I needed. I was convinced that the patterns in music would translate to numeric patterns when digitized, which would become visible when processed trough my drawing application.

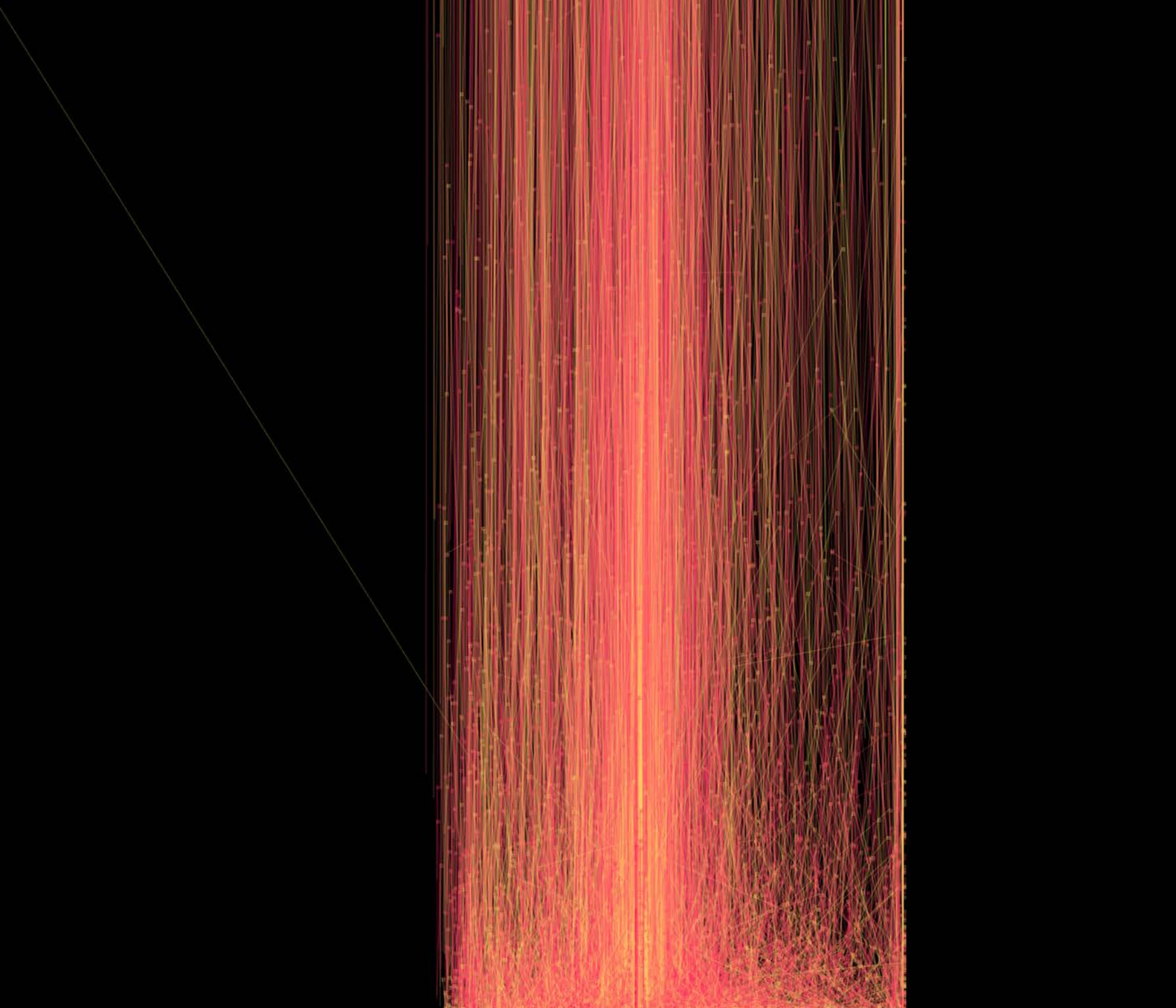
I have always enjoyed John Coltrane's music and decided to use his composition *Giant Steps* for my experiment with drawing music. I tried to export the entire song into ASCII format and crashed my computer. Realizing that it was just too much data, I cut it in half, and crashed my computer again. I proceeded to cut the song into smaller and smaller chunks until I found a manageable chunk: the first five seconds. I dropped the data into my drawing program and a beautiful composition came out.

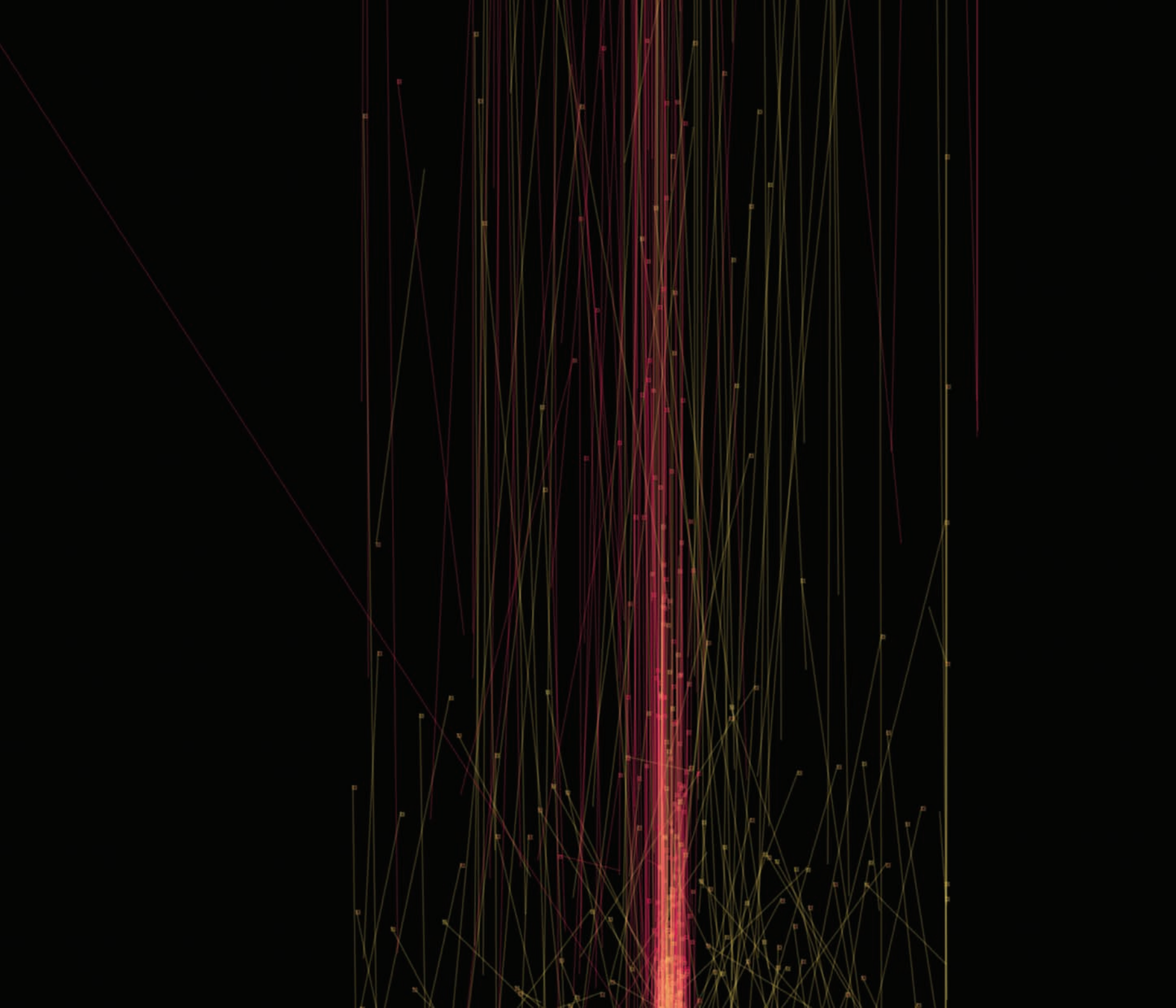
I knew that five seconds was likely not enough time to establish much of a pattern in the song, but it didn't matter to me as I had proved to myself that I could write a song drawing tool by taking advantage of the common language across digital media of ones and zeroes.

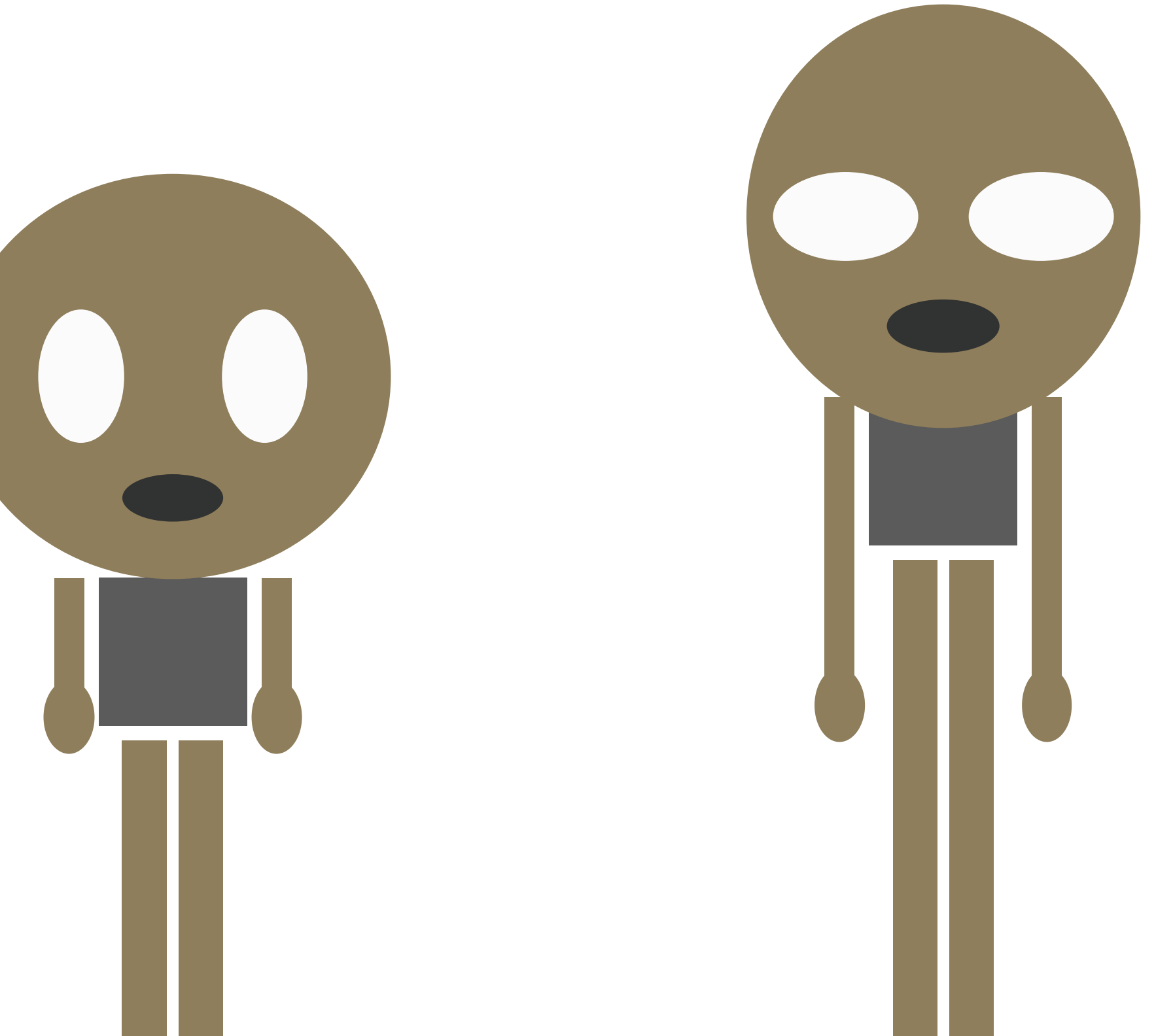
While I was happy to have visualized the data from the John Coltrane song, I wanted something that I had more control over: less of a visualization project, and more of a drawing project. What if I produced the music live and created the drawing with real-time audio data?

I play several instruments equally poorly. At different times in my life I have tried to learn the clarinet, drums, and harmonica with varying degrees of success. For my live music drawing program, I settled on the harmonica, as it was the most portable. I had a little experience messing around with krister.ees, an audio library for Processing, so I decided to utilize it for this project. I hooked up my microphone, started writing code, and eventually worked through a tutorial that created real-time waveforms responding to the data captured from my harmonica. While I was pleased to have completed the loop of real-time audio input to visual output, the visualization it self was still just a boring, run-of-the-mill waveform. So I started to adjust the code to try and make the visuals feel like the audio sounded. I made adjustments to color, motion, line thickness, and every other variable I could find until the visualization organically evolved into a visual equivalent for my sounds. No longer just a series of waveforms, I was now drawing with my harmonica.

Harmonica Visualization #2 ▸
Processing, 2009







Algorithm

I've always found libraries difficult to navigate, and to this day I don't really understand the Dewey Decimal System. As a child, I would just wander the stacks, reading the spines and then extrapolate the theme of any given section.

Because I could not understand the card catalog, it became important for me to memorize where the good sections were.

To this day, I have a pretty crisp image of the two shelves of books that lined the back left corner of the Billerica Public Library. One row deep on the right side and one shelf up from the floor were the children's drawing books. I remember the drawing books were sandwiched between other books about juggling, ventriloquism, magic tricks, and karate. I am sure there was some Dewey-type-logic governing this seemingly disparate cluster of books, but to a 6 year old it was just the "awesome section."

Among this awesome assortment of books, none was more awesome than the drawing books of Ed Emberly. What child could resist books with titles like *Make a World* and *The Drawing Book of Weirdos*? The covers of these books were littered with a menagerie of curious colorful creatures all created from just a few simple shapes. Although I could not yet read, I was able to intuitively understand the visual instructions, the step-by-step accumulation of simple visual components, combining a line here and a circle there, to create a much larger and more exciting whole.

Emberly used slow disclosure, simplicity, and structural transparency to inspire millions of children to draw. The real lesson of Emberly was not in memorizing the specific algorithm of how to draw a witch or dog, but in understanding that you could draw anything, no matter how complex, by disassembling and abstracting it to its fundamental parts.

Like most 6-year-old boys, I was also excited by the idea of superheroes and action figures. In addition to crayons and drawing books, I received G.I Joe and He-Man figures for birthdays and Christmases.

These action figures allowed me to stage epic fantasy battles, but I still wanted to generate characters of my own. So I created a team of superheroes called the Wee

◀ *Big Two*
Processing, 2008

Wee Men. I know this sounds ridiculous, but I was 6 and this seemed like a perfectly suitable name at the time.

Initially, there were three Wee Wee Men, each one comprised of two simple overlapping shapes. Think of a square in your mind. Now remove the top line. It should look like a staple. Now duplicate this shape and flip it vertically, and move it down slightly. It forms a rectangle with two lines coming off the bottom and two lines protruding from the top. The lines extending from the top were arms, stretched straight up in the air, the lines shooting downward were the legs. In the middle, where the horizontal rectangle was formed, I would add two dots for eyes and a crescent for a mouth. The other two Wee Wee Men were of similar construction but used triangles and half-circles.

I eventually realized that by cross-pollinating half-circle with square, square with triangle, etc., I could expand to a much larger cast of characters. I had devised an algorithm for drawing superheroes and I spent many hours drawing the scenery with mountains, plains, and rivers, and then populating them with Wee Wee Men societies playing out different situations, usually gory battles where someone would lose a leg or an arm.

Later in life, I made my way into the work world and found little time to create artwork for pleasure. In one of those rare moments where I would create my own work for pleasure, I started a series of algorithmic drawings in Maya 3D software. I had taught myself MEL script, the native programming language for Maya Software. The desire to explore how I could apply my new scripting knowledge to create complex, abstract 3D drawings was consuming me. I would wait until after my wife and I went to bed, get back up, and write 3D drawing algorithms.

I was addicted to the surprises of this process. The first surprise would come after I created a simple algorithm and ran the routine. Although driven by math, these drawings were strangely organic, each one a surprise. The second surprise would come after setting up the virtual camera and lights on the model and rendering out the high-quality image. Depending on the complexity of the model, the render could take several hours before finishing. Sometimes the results were stunning, other times they were relatively unremarkable, but it was the variety of results and the lack of complete control that made the surprises so rewarding.

Case Study: Player Visualizer

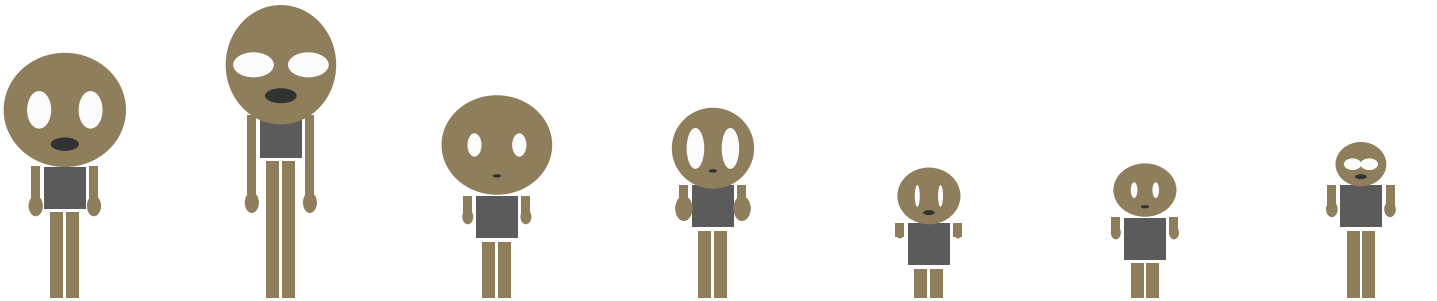
The sweat from my face poured down onto the keyboard. Even in the shade of the wooden cyber-shack, the heat of St. John in early September was unbearable. I had snuck away from my new bride on our honeymoon to research any off-season trades that may have affected my fantasy basketball draft strategy. The season didn't even start for another month. We had decided to spend the week at a campground tucked into the hills on the shore of a beautiful Virgin Islands beach and here I was paying a dollar a minute for Internet access to catch up on basketball news. The pleasure I received from watching basketball had been eclipsed by my fascination with basketball statistics.

This was strange on many levels. Math was responsible for the majority of my scholastic low-points, including failure to place into a credit-bearing course in college despite spending a month in summer school re-taking algebra. This obsession with statistics could really only be attributed to one thing: sibling rivalry.

Like most siblings, my younger brother and I have always been pretty competitive with each other. Unlike most siblings, our competitions almost always played out in some obscure form: Frisbee golf, eating hot foods, or statistical analysis, for example. We spent a lot of time reading and sharing literature on basketball statistical analysis including *Basketball on Paper* by Dean Oliver, and Dave Berri's *Wages of Wins*. Our annual fantasy basketball league was our chance to pit our knowledge and theories against one another; it had become an obsession for us both.

Statistics were interesting to me for more than just bragging rights. As a designer creating courtroom graphics for an accident investigation company, I tried to represent complex data in a graphical form that would be intuitive for a jury to understand. This job helped me realize that I enjoyed the process of distilling complex information into simple graphics. That career provided me a never-ending stream of data to work with: crush-data derived from vehicle collisions, thermal maps of building fires, and the fatigue-testing data of micro electro-mechanical systems to name a few. But after leaving that job, there was a shortage of interesting data to visualize. Basketball statistics filled that void for me.

All sports are algorithmic in nature, some more than others. For example, statisticians love baseball because it is arguably the most algorithmic of the more popular sports, making it the easiest to model. Baseball has the longest season, 162 games, which provides a large statistical sample. The action is slow and isolated, making



it easy to document. Success and failure is measured in discrete steps based on which base a player gets to: first, second, third, or home. Statistically speaking, basketball represents a much more dynamic space than baseball. It features continuous and decentralized action, making it a challenging sport to document and analyze statistically.

So when DMI guest lecturer Peter Kirn asked us to choose a data set to visualize for a summer course in Processing, basketball statistics were the obvious choice. The first step was to figure out how to import data. I took the information for a single player, in a single game, cleaned it up in Excel, and figured out how to import it into Processing. After tracing the data to the screen, I figured out how to map it to the length of a rectangle, approximating a crude bar graph. I was confident that I understood how the process of importing data worked, but I wanted to make something more interesting than a bar graph. I thought about what my visualization would look like if I stayed with this bar graph approach, but tried to represent every statistical category for each player for the entire previous season. I realized that if I continued on this path, the resulting visualization would be overwhelming and illegible. This is exactly the kind of visualization problem I relish.

I started mapping the same data to other shapes: circles, triangles, lines. Then, inspiration struck: I could write an algorithm to draw caricatures of the players using their statistics. It was a similar process to the one I used to draw the Wee Wee men super heroes. I could combine simple shapes to develop a wide variety of characters. What made this especially appealing was that many statistics could be tied to human physi-

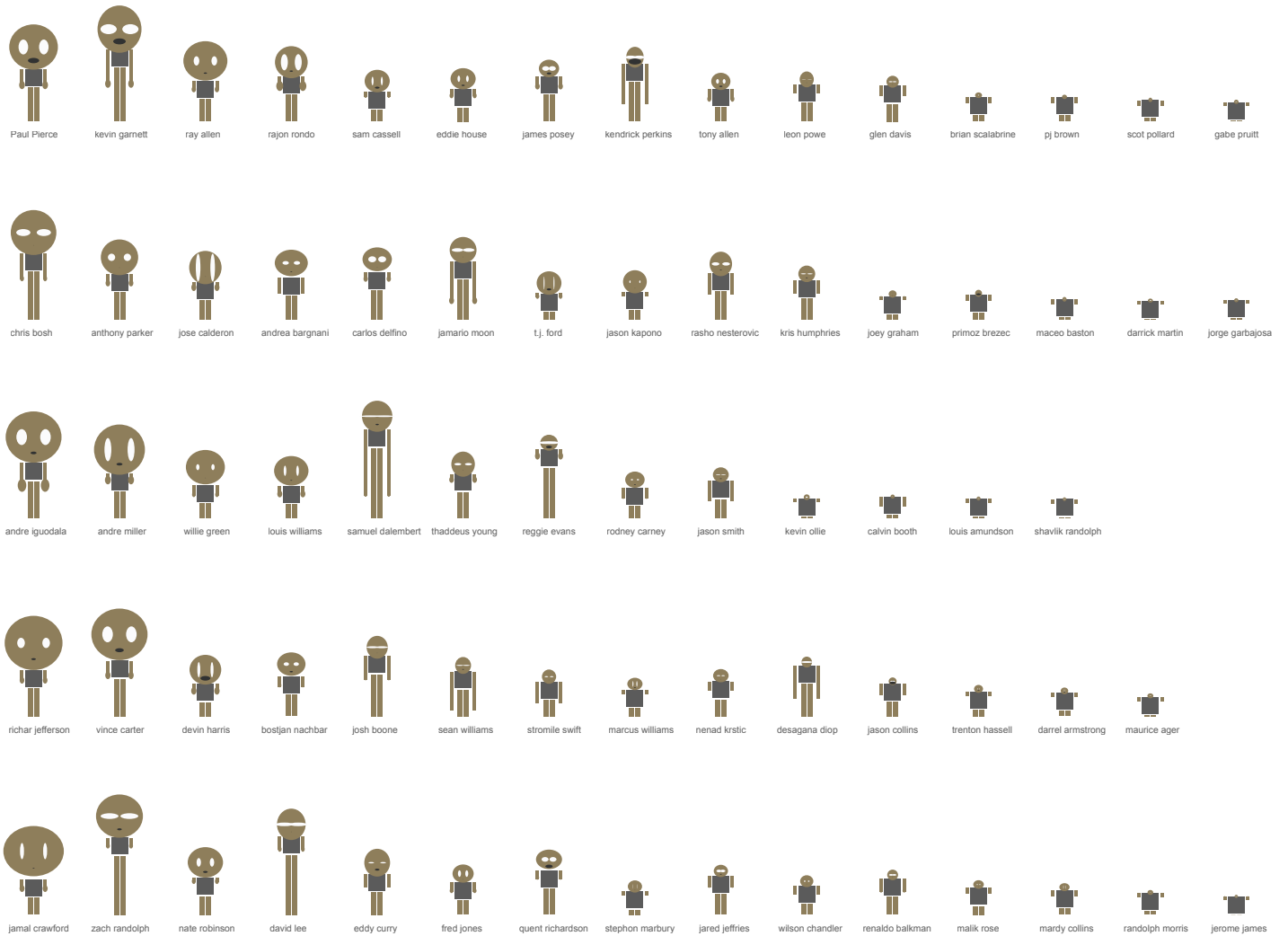
Celtics
Processing, 2008

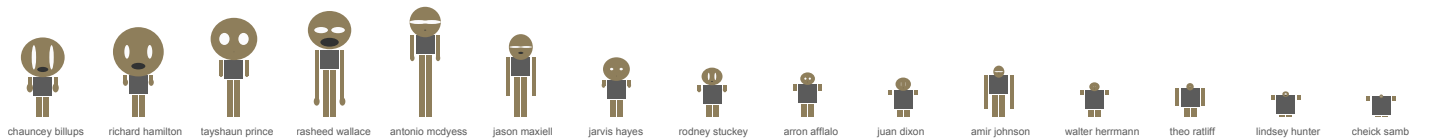
cal attributes. For example, a player with great passing ability has good eyes, a rebounder typically is taller with longer legs, shot blockers often have longer arms, etc. This brought on the type of excitement that gets me past the technical challenges of learning a new programming language.

I worked feverishly for several nights perfecting my caricature drawing algorithm for a single player. When I felt like I had a decent likeness of that player, I imported the data for the rest of the 432 players taken from the 82 games played that season and ran the program. I realized, like never before, the power of the computer as a drawing tool. With a few simple code tweaks, I had gone from drawing a single player to drawing 432, and many of them looked like the actual person they represented. More importantly, it was incredibly easy to make quick visual comparisons between the players and even the teams and extract the statistical differences. Instincts provide us with an ability to read the human form incredibly fast and notice subtle differences without difficulty. Although I knew this somewhat instinctively, I confirmed it when my classmate Scott sent me information on Herman Chernoff. In 1973, Chernoff invented a visualization technique called the "Chernoff Face" which acts on this same basic principle using subtle changes in simple hu

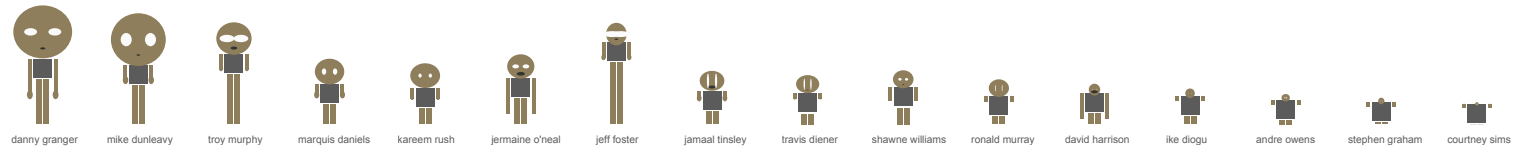
man caricatures to represent large data sets.

Drawing just one of these basketball caricatures with statistical accuracy using a pencil or even Adobe Illustrator would have taken me an hour. Now, as if by magic, I had a machine that could generate thousands.


Paul Pierce
kevin garnett
ray allen
rajon rondo
sam cassell
eddie house
james posey
kendrick perkins
tony allen
leon powe
glen davis
brian scalabrine
pj brown
scot pollard
gabe pruitt
chris bosh
anthony parker
jose calderon
andrea bargnani
carlos delfino
jamario moon
t.j. ford
jason kapono
rasho nesterovic
kris humphries
joey graham
primoz brezec
maceo baston
darrick martin
jorge garbajosa
andre iguodala
andre miller
willie green
louis williams
samuel dalembert
thaddeus young
reggie evans
rodney carney
jason smith
kevin ollie
calvin booth
louis amundson
shavlik randolph
richar jefferson
vince carter
devin harris
bostjan nachbar
josh boone
sean williams
stromile swift
marcus williams
nenad krstic
desagana diop
jason collins
trenton hassell
darrel armstrong
maurice ager
jamal crawford
zach randolph
nate robinson
david lee
eddy curry
fred jones
quent richardson
stephon marbury
jared jeffries
wilson chandler
renaldo balkman
malik rose
mardy collins
randolph morris
jerome james

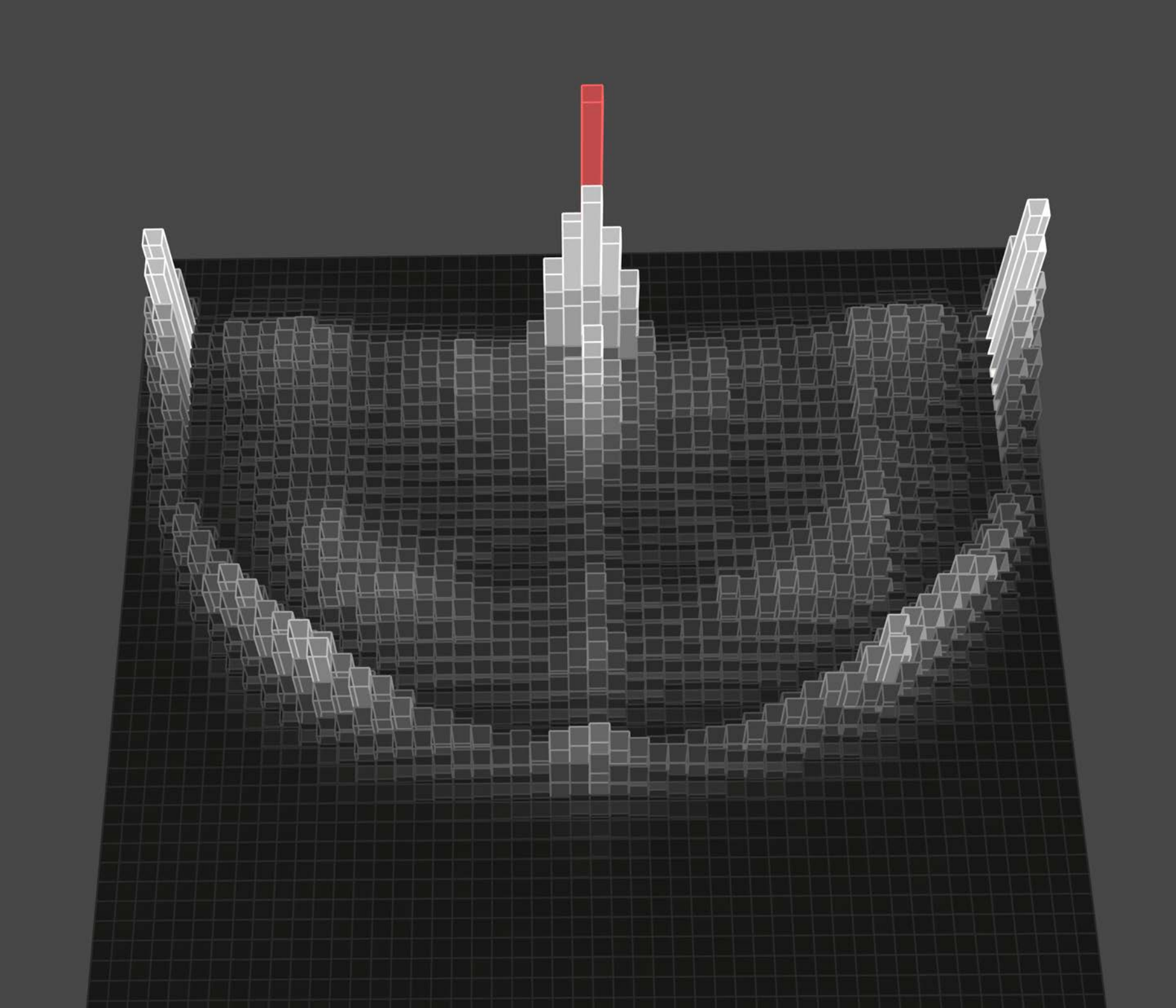

chauncey billups
richard hamilton
tayshaun prince
rasheed wallace
antonio mcdyess
jason maxiell
jarvis hayes
rodney stuckey
arron afflalo
juan dixon
amir johnson
walter herrmann
theo ratliff
lindsey hunter
cheick samb


lebron james
zydrun ilgauskas
wally szczerbiak
joe smith
daniel gibson
devin brown
delonte west
damon jones
sasha pavlovic
ben wallace
anderson varejao
dwayne jones
eric snow
billy thomas
lance allred
kaniel dickens


danny granger
mike dunleavy
troy murphy
marquis daniels
kareem rush
jermaine o'neal
jeff foster
jamaal tinsley
travis diener
shawne williams
ronald murray
david harrison
ike diogu
andre owens
stephen graham
courtney sims


ben gordon
andres nocioni
luol deng
kirk hinrich
larry hughes
drew gooden
tyrus thomas
joakim noah
thabo sefolosha
chris duhon
aaron gray
shannon brown
adrian griffin
viktor khryapa
thomas gardner
demetris nichols
cedric simmons


michael redd
mo williams
andrew bogut
charl villanueva
desmond mason
yi jianlian
bobby simmons
charlie bell
royal ivey
dan gadzuric
ramon sessions
jake voskuhl
awvee storey
michael ruffin



Distribution

The geek culture blog Boing Boing recieves more than ten million page views per month. On May 11th of 2009, Boing Boing Gadget, a sister blog to Boing Boing, published an interactive data visualization I created with my younger brother.

The visualization displayed a 3D bar graph of every shot taken in the NBA for the 2008 season, positioned relative to the location on the court at which the shot was attempted. We had seen a heat map graphic that used color to depict the information in 2D, but felt like it did not really communicate the discrepancy of shot volume between the locations on basketball court from which the shot was attempted. (Witus) We decided that a navigable 3D interface could enhance the discrepancies between shots while increasing the clarity of shot location.

The visualization had been on my blog for seven months. I had no reason to believe it would be treated any differently than any other post. Typically one or two of my classmates would read my posts, and occasionally my brother, but that was my about it. Being the lazy blogger that I am, I did not realize that Boing Boing had picked up my post until several months after I posted the article to my blog.

I was surprised and delighted to find that many other blogs and websites found our visualization interesting.

There was Alark Joshi, the post-doctoral student from Yale who was doing work on open-source medical imaging and had developed ground-breaking visualizations of hurricane data. Alark posted a large image of our graphic on his "visualization blog" with the comment, "An awesome visualization of all the shots from the NBA 2007-2008 season." (Joshi)

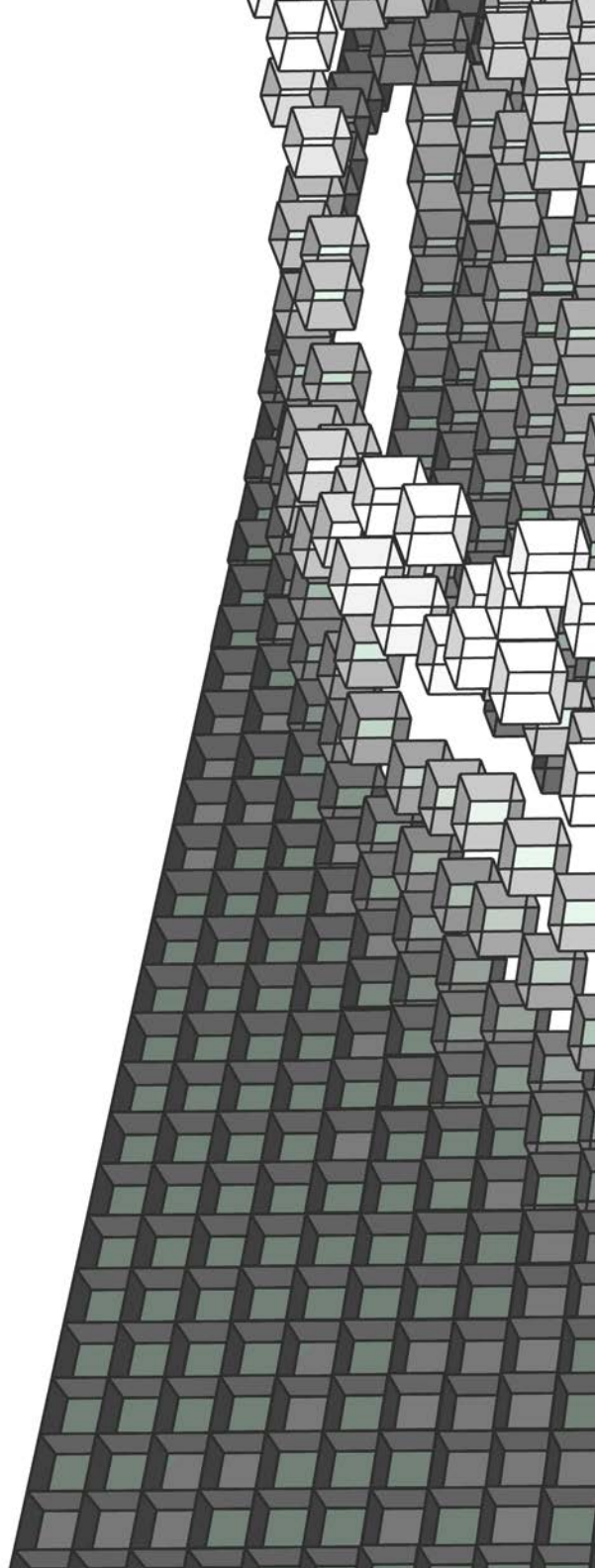
AERS, a company specializing in "advanced transactional data processing," was extremely complimentary. They posted the visualization to their website with the title, "Fantastic 3D visualization in Sports" and finished the article by stating "Kudos to Jason… hope your phone is ringing off the hook with NBA and NHL teams." (AERS) No one ever called from the NBA or NHL. But as of January, 27, 2010, the AERS reposting of our visualization project had 1,478 views.

◀ *Shot Distribution*
Processing, 2008

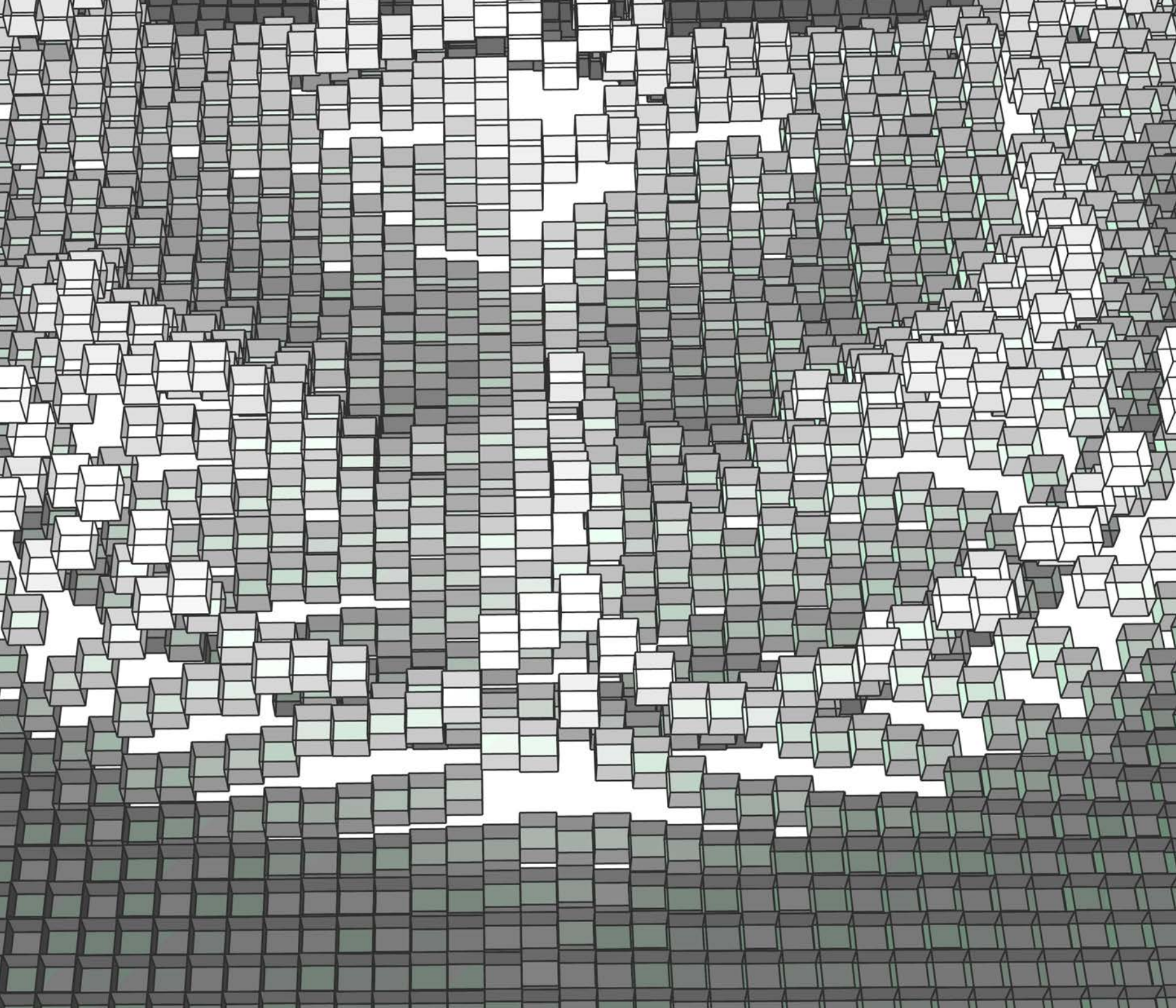
Not all the feedback was complimentary. Among the comments on Boing Boing's original posting were, "It's a tongue. I can't be the only one who sees this, " and, "That image reminds me, very strongly, of the boss fight with Andross in StarFox." Unbeknownst to us, our project had developed a life of its own.

Of all the drawings, paintings, and illustrations that I have created in my life, I never thought that a chart of three-point shots would get the most exposure and generate the most interest. This is a uniquely digital phenomenon with little or no analog equivalent. Before working on the computer, like most artists and designers, I would create work, show my friends and family, and every once in a while put work in a local gallery where maybe 50 to 100 people would see it. The Internet allowed us to put our work in front of more eyes and inspired more feedback than all my other work combined.

Not to imply that it is simple to get people to care about your work. The visualization had laid dormant on my blog for months before being referenced by a more popular blog, Boing Boing. The Internet, with its 16 billion users gives anyone with a computer and a modem access to an enormous audience. The chances of finding several thousand people who care about basketball visualizations in my everyday life are pretty slim. However, on the Internet, subcultures of people who share my fascinations become a readily available audience.



Shot Distribution Detail ▸
Processing, 2008



Conclusion

Every fall it is a DMI tradition for current and past DMI students to share "pearls of wisdom" with incoming students. In the past two years, I have given the same advice both times: "get to know your classmates as fast as possible."

The followup to this "pearl" is to "go out after class with and drink beer and eat pizza." I actually think this second part is just as important as the first, or perhaps it is just the best way I know for people to get comfortable with each other. The DMI faculty have consistently done a great job of choosing an eclectic group of design professionals with expertise varying from architects, filmmakers, cermacists, writers, to usability experts, performance artists, and industrial designers from all around the globe. I made it a point to meet every student that entered the program, learn their name, their background, and invite them out for beer and pizza. I wasn't just being nice. I was being selfish. Every time I met a new student, I learned something new about design and I made a new friend.

One of the primary reasons for applying to graduate school for me was the hope that I would become part of a group of creative professionals who would share my interests and help me develop new ones. My biggest regret for my time spent at DMI is not having chosen to work on projects as a group more often.

Programming, like drawing is often an isolated activity. I spent 90 percent of my three years at DMI locked up in my home office working into the night by myself. There were certainly times where I enjoyed working by myself. Drawing and more recently programming have provided valued "me time." About halfway through my time in the program I was working on a project that involved blowing on a balloon tied to a wii-mote to navigate Google earth. I spent a lot of time thinking about how to make this fun for the user. At one point my wife and I tied 30 helium balloons to our wii mote and blew it around the backyard with a leaf blower. The balloons carried the wii mote high up into the air sailing toward the neighbor's yard and eventually into a tree. Although this approach could be seen as a failure, we had a fantastic time documenting it and trying to get the wii mote down from the tree. Several months later I tried to give a

◀ *Age 5, Self Portrait*
Crayon, 1983

live demo with a single balloon carefully placed on a balancing wii mote such that it would rock back and forth steering through Google earth when the user blew on the balloon. During the end of the semester reviews, I asked everyone to move into the room next door to interact with my demo. When I tried to start my demo, the laptop had fallen asleep and the demo would not work. Even if the demo had worked, it was not really that interesting. It was too predictable, too controlled, too much of a canned experience. The making process was more fun with other people and ultimately more fun then interacting with my final project.

The semester after the failed balloon demo, I took Gunta Kaza's design as experience class. It was good timing. Gunta's assignments put more emphasis on the experience, interaction, and collaboration than they did on the artifacts resulting from experience. In one project, we needed to design a solution for characters in conflict from one of several selected stories. I selected an excerpt from *The Life of PI* detailing several days in the lives of a starving man and a tiger stranded on a raft in the ocean. This seemed like an absurd situation so I came up with an absurd solution. I decided that it could be a friendly tiger but the inability to communicate would keep them from realizing they meant each other no harm. I designed a helmet that projected cartoon speech bubbles, allowing them to communicate in a way which would be clear and non-threatening. I thought I was done with the project, but Gunta asked us to push further. Have solved the immediate crisis, I decided they needed a way to pass the time. Keeping with the absurdist solutions, I decided that Pi and the tiger would write a book of haiku together which they could then share with the world when they were saved. I had wanted to involve more people in my projects, so I decided to ask my family, friends, and classmates to write a haiku from the perspective of one of the characters. I was surprised by how willing everyone was and how much fun everyone had, and was further inspired to involve others in the creation of future projects.

A few weeks before finishing this thesis book, I had realization. I had learned many new things at DMI, done a massive amount of research, and developed many new skills, but in the end I returned to drawing. I thought about the other DMI graduates and realized that most of them had returned to their areas of expertise as well. Whether returning to writing, architecture, music etc., we almost always come back to our areas of expertise, but with an entirely new viewpoint. I would recommend to future DMI graduates who are struggling as I did to find that big thesis idea to remember where they came from. I think that is where the thesis lives.

Pi Haiku
Cardboard, 2008

Bibliography

AERS – advanced ecommerce research systems, Fantastic 3D Visualization in Sports. http://www.researchadvanced.com/2009/05/fantastic-3d-visualization-in-sports/ (accessed 12 April 2010)

Croquet, Croquet System Overview,
http://www.opencroquet.org/index.php/Croquet_SDK

Chipp, Herschel, B, *Theories of Modern Art: a source book by artists and critics* (University of California Press(1984), Paperback, 680 pages)

Joshi, Alark, "Visualization in Sports", Visualization Blog. http://visualizeit.wordpress.com/2009/05/07/visualization-in-sports/ (accessed 12 April 2010).

Moggridge, Bill, *Designing Interactions* (The MIT Press (2007), Edition: 1, Hardcover, 766 pages, 2007)

Fry, Ben, *Processing: A Programming Handbook for Visual Designers and Artists* (The MIT Press (2007), Hardcover, 736 pages)

Reas, Casey, Software & Drawing, http://artport.whitney.org/commissions/softwarestructures/text.html (accessed 12 April 2010)

Resnick, Mitchel, *Turtles, Termites, and Traffic Jams: Explorations in Massively Parallel Microworlds* (Complex Adaptive Systems) (The MIT Press (1997), Paperback, 181 pages, 1997)

Salen, Katie, *Rules of Play: Game Design Fundamentals* (The MIT Press (2003), Edition: illustrated edition, Hardcover, 688 pages, 2003)

Wardrip-Fruin, Noah, *Expressive Processing: Digital Fictions, Computer Games, and Software Studies* (The MIT Press (2009), Hardcover, 480 pages, 2009)

Ware, Colin, *Visual Thinking for Design* (Morgan Kaufmann (2008), Paperback, 182 pages, 2008)

Witus, Eli, "More Shot Charts – Where Players Score Points – eFG% by Location", Count The Basket, Advanced Stats for Basketball. http://www.countthebasket.com/blog/2008/03/27/more-shot-charts/ (accessed 12 April 2010).