AutomaTiles

Tangible Cellular Automata for Playful Engagement with Systems Thinking

by

Jonathan Bobrow

B.A. University of California Los Angeles, 2008

Submitted to the Program in Media Arts and Sciences, School of Architecture and Planning,
in partial fulfillment of the requirements for the degree of
Master of Science in Media Arts and Sciences
at the
MASSACHUSETTS INSTITUTE OF TECHNOLOGY
June 2016

Signature of author:________________________________

Jonathan Bobrow
Program in Media Arts and Sciences
May 6, 2016

Certified by:________________________________

Kevin Slavin
Assistant Professor of Media Arts and Sciences
Thesis Supervisor

Accepted by:________________________________

Pattie Maes, Ph.D.
Academic Head
Program in Media Arts and Sciences

AutomaTiles | Tangible Cellular Automata for Playful Engagement with Systems Thinking

by

Jonathan Bobrow

Submitted to the Program in Media Arts and Sciences,
School of Architecture and Planning, May 6, 2015 in Partial Fulfillment of the Requirements for the Degree of Master of Science in Media Arts and Sciences

Abstract

There is an increasingly vital awareness that our world is an aggregate of complex systems, emergent behavior, and system dynamics. The perceptual and analytical tools for exploring and studying these systems, however, have generally been relegated to scientists (whether mathematicians, physicists, biologists, economists, or computer scientists). Thus, as more and more people become aware of such systems, most people are still excluded from engaging with complex systems.

By inventing a new tool and interface, consisting of playful objects called AutomaTiles, I propose a new approach for fostering a more aware society of systems thinkers. AutomaTiles provide a three-tiered approach to making systems thinking more accessible. Firstly, AutomaTiles are a friendly and approachable set of playful objects; seen simply as toys, they afford the surprising effects of emergent behavior when brought together in aggregate. Secondly, AutomaTiles can be a tool for exploring collective behavior, distributed algorithms, and models of systems (whether forest fires or social phenomena) from a hands-on perspective. Lastly, AutomaTiles are a new kind of platform for games, bringing computational intelligence to table-top games, bringing together the social dynamics of face-to-face interaction with the complexity afforded by conventional video-games. Expanding the work on the future of board games from Playful Systems, we have created a novel digital-physical interface for playing games that allows for modes of gameplay never before possible in a table-top game.

This thesis will illustrate the design decisions and affordances of AutomaTiles as a platform for engaging with these three tiers of the exploration and manipulation of complex systems.

Thesis Supervisor: Kevin Slavin
Title: Assistant Professor of Media Arts and Sciences

AutomaTiles | Tangible Cellular Automata for Playful Engagement with Systems Thinking

by

Jonathan Bobrow

The following served as a reader for this thesis.

Radhika Nagpal, Ph.D.

Core Faculty, Wyss Institute Professor

Harvard SEAS

AutomaTiles | Tangible Cellular Automata for Playful Engagement with Systems Thinking

by

Jonathan Bobrow

The following served as a reader for this thesis.

Mitchel Resnick, Ph.D.
Professor of Media Arts and Sciences
MIT Media Lab

Acknowledgements

This thesis would not have been possible without my advisor Kevin Slavin, my Playful Systems group mates, Che-Wei Wang, Taylor Levy, Julie Legault, Kamal Farah, Greg Borenstein, Shoshana Tekofsky, Miguel Perez, Mike Lazer-Walker, and Benjamin Berman. I would also like to thank my readers, Mitchel Resnick and Radhika Nagpal, whose literature influenced me long ago and for their support and critical suggestions from the start of the process.

I would like to thank the Undergraduates who assisted in the realization of AutomaTiles firstly with Claire Patterson, who modeled the first cases to contain the electronics and Joshua Sloane, who creatively solved problems and shared my excitement from the beginning with the possibilities of AutomaTiles as a new platform for discovery and play.

The Media Lab is a beautiful garden of minds, and I consider myself lucky to have been planted so generously there. This list of names is far from complete, but the following people have had a constant presence in the work, from small play tests, to whiteboard sessions or late night hacking. Here is a list of minds I feel indebted to:

Amanda Ghassaei	Joi Ito	Neil Gershenfeld
Chikara Inamura	Shantell Martin	John Difrancesco
Palash Nandy	Marko Ahtissari	Tom Lutz
Tomer Weller	Linda Peterson	Kristina Bonikowski
Jie Qi	Keira Horowitz	
Greg Leppert	Jessica Tsymbal	
	Kevin Davis	

If not for the constant support of my partner, Jamie Tsukamaki, from molding or casting parts, to assembling or even simply playing, I cannot imagine realizing my vision with such clarity. With critical feedback and a killer Tortilla soup, Jamie helped keep my momentum through the rollercoaster MIT calls grad school.

In the complex system with which I grew up, my family has always been there to send love and support through the best and worst of times. My mother Susan, sister Jennifer, brother Adam have all contributed to and embraced my curiosity from a young age, and for that I am grateful.

For my dad.

I would like to dedicate this thesis to my father, coach and role model for life, Jerry Bobrow. He inspired my love of mathematics, my constant drive to learn from as well as teach others, and what it meant to be part of a team. As an author of far more valuable (or at least widely distributed) literature, he would have delighted in sharing this period of my life with me and so I wish to share this thesis in his name.

I was lucky then and I am still lucky now.

Table of Contents

1. Introduction

While the HCI community has made strides in how we communicate with one another and augment our experiences through technology through new approaches and effects to programming, in 2016, cutting edge engineering engages problems and opportunities of far greater complexity. For example, Kevin Esvelt's work in "sculpting evolution" and the coming revolutions in synthetic biology require consideration and thoughtfulness around all the ways that evolved organisms, communities, and ecosystems interact. In time, as we have come to engage computation systems casually, we may soon do the same with far greater scope, complexity, and stakes.

Systems are defined by their interconnected parts resulting in feedback loops and emergent properties or unexpected outcomes from an aggregate of individually understood components. My interest here lies in how people gain an intuition for this complexity, engage with emergent behavior in the kind of "magic circle" provided by toys and games. I have therefore developed a platform for users to engage with these dynamics and develop a language for playing with systems.

I am hardly the first to approach this subject; artists and engineers alike have long been fascinated by the kind of emergent properties found in even the simplest of systems. Not only have artists and engineers engaged these systems and defined aspects of them for their work, they have also sought to build platforms for engagement. I will survey many of these technologies, as well as how people have engaged with them and then expand upon my explorations for a new platform, one that has been built from the ground up as both a study in design as well as interaction.

My own advisor, Kevin Slavin, recently published about the transition from user centric design to engaging with the ever increasing complexity as participants. Design for participants speaks to a very important difference in the way people need to design moving forward. We have moved beyond cradle to cradle design, considering the life-cycle of a product, now considering how an object will coexist during its lifetime. What kind of influence will an object have? Will introducing a new element into a system have consequential effects or will the system maintain equilibrium? What are the thresholds and how will we know when we've gone too far?

John Conway looking at an early CRT displaying his iconic Game of Life (1972)

2. Background

Complex systems have a long history in human thought. A Sufi tale of blind men trying to describe an elephant is one common example how the whole is different than the sum of its parts. In this section, I will first consider the scientists' approach to complex systems and a call to action for better understanding, introduce artworks that have been personal inspirations, and summarize precedents in making objects to think with.

2.1. The Scientists

In 1956, Jay Forrester started a new group at MIT, a group focussed entirely on complex systems, building models, and analyzing data to tell stories about the many complex systems that surround us and that we are part of. Members of his group published a book in 1972 called the *Limits of Growth*, which looks at a computer model of populations and economics and determined that we were headed in a negative trajectory. Even though every model — computational or not — is limited in its abilities to simulate a system, there was validity to their publishings and concern. In fact I became aware of the book because of an article published in The Guardian around the time I was entering the Media Lab. The article, entitled *Limits to Growth was right*[1] charted current data with that predicted 40 years ago and found that the models were not (yet?) incorrect.

Before computers were accessible, Game of Life was simulated on a Go board, it's first physical instantiation.

The two main figures behind *Limits to Growth* were Donella and Dennis Meadows, a husband and wife duo that contributed a lot to the field of systems thinking. In fact, before Donella's passing, she started a foundation and nearly completed a book on just that, *Thinking in Systems*. It was Donella who pointed out so bluntly that from systems arise emergent behaviors, but not all emergent behaviors are good: in fact the world's biggest problems "hunger, poverty, [and] war" are all unintended consequences. No single person on earth wants these problems to persist, but despite our best efforts, they continue to plague our planet (Meadows 4).

> **"The solutions are in our hands, [but] we must *do things*, or at least *see things* and *think about things*, in a different way."**
>
> *- Donella Meadows, Thinking In Systems*

The only way I could read this was as Meadow's call to action. These problems are too important to not do something about, the world needs more systems thinkers, and system thinkers need tools to think with.

---

[1] http://www.theguardian.com/commentisfree/2014/sep/02/limits-to-growth-was-right-new-research-shows-were-nearing-collapse

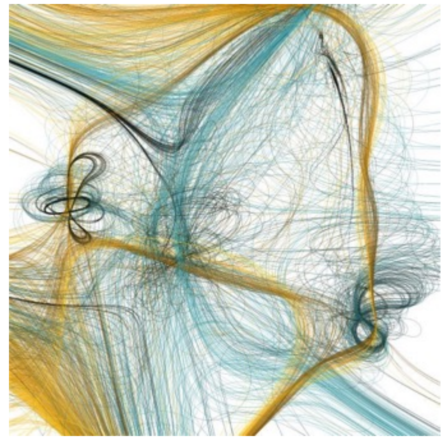
Just as Meadows was writing about systems thinking, John Conway, a mathematician was searching for the simplest set of rules to create unpredictable behavior. That sounds counterintuitive, but that is precisely what emergent behavior is; a surprising organization or disorganization from a seemingly controlled environment.

> **“That mix of order and anarchy is what we now call emergent behavior.”**
>
> - *Steven Johnson*

Conway called his ruleset ‘life’ and it soon became known as Conway’s Game of Life. The simulation game could be played on graph paper or a checker board by treating each square or cell of a grid as its own organism following three simple rules:

- Survival - if in the company of 2 or 3 living neighbors, remain alive to see another day.
- Death - if in the company of less than 2 neighbors, starve of isolation, if in the company of more than 3 neighbors, die of overpopulation.
- Birth - if surrounded by 3 living cells be born of ‘trisexual’ reproduction.

The (almost) organic patterns captivated the masses and as computers became accessible, inspired scientists and hobbyists to try their own initial conditions, discover what emerges, and search for longer lived or more complex behaviors.
The *Game of Life* reaches a broad audience of science and math hobbyists, will be referenced throughout the thesis since it served as a starter's block, from where to push off of. The humbling application of Occam's razor for rulesets is a model that becomes core AutomaTiles architecture.

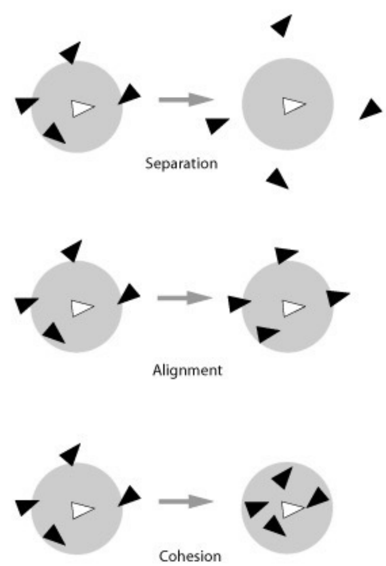


Path, Casey Reas (2001)
Based on synthetic neural systems, each line describes the history of a single system as it navigates its environment

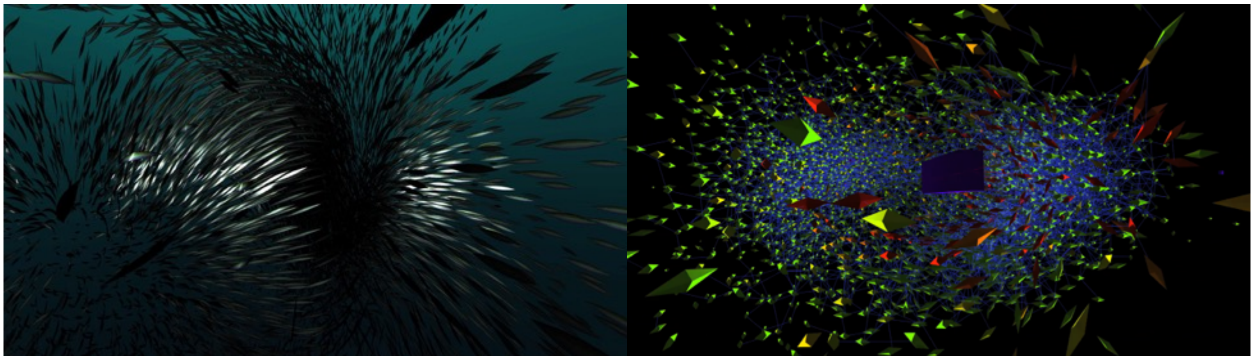
2.2. The Artists

With the advent of computation as a tool for artists, the communication of complex systems and their effects has leaked out of the lab into culture itself. Before computers were a household item, the ideas of computation inspired artists as well. The work of Sol Lewitt reveals art through a series of instructions, and according to many, the art is the instructions themselves. Braitenberg vehicles, simple machines taking on characteristics we associate with sentient beings inspired the generative artwork of Casey Reas, which explores simple rulesets building complex imagery. The mathematician, John Conway, and his Game of Life inspired me among many others to play with cellular automata as an art-form, and the work of Vi Hart and Nicky Case expands to a new form of art, giving rise to rich stories through cellular automata. Here are some of the artists whose work I admire and has served as an inspiration for the aesthetic and ethereal properties of AutomaTiles.

The way that animals gather, corral, and sometimes ebb and flow can be referred to as flocking behavior — it is mesmerizing at first glance. How can such beautiful forms arise without a conductor, a leader, or prior coordination? In 1986, Craig Reynolds developed an algorithm for computer based animation of these emergent behaviors and called the model Boids. Used for simulating birds, fish, and other swarm behaviors, media artists began to adopt these algorithms to introduce an orderly form of chance.



Three simple rules that each *Boid* observes with its local neighbors to result in flocking behavior. (Craig Reynolds, 1986)

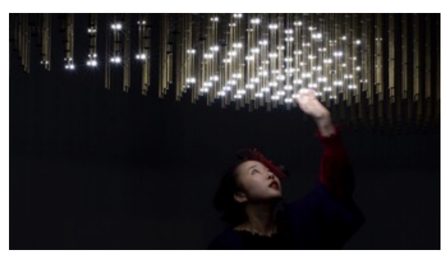


Boil Up by Robert Hodgin. The left image shows an early still frame from a real time flocking behavior based on Craig Reynolds' Boids. The right side is a debug view, but how Hodgin imagined the aesthetics originally abstracting the fish and showing their behavior through convincing animation. (2013)

Robert Hodgin, formerly of the special effects industry began to play with Boids and similar algorithms to create stunning and complex imagery, too difficult to animate by hand. The emergent properties result in unique artwork every time the application is run, much like setting a flock of birds free to paint a picture. Seeing Hodgin's work years ago left a lasting impression on me. While he presented to an audience at Eye-O Festival, it was clear that he was enjoying his work, treating it as play, and that the results were surprising him as much as it was surprising us.

In addition to the fluid motion provided by the emergent properties, there was attention to detail of color palettes, shading, glow, and the ephemeral properties that made the artwork resonate. AutomaTiles needs to be a tool that delights and surprises me as well as its less familiar users, so we can play and explore the affordances of living devices. Each of these fine tuned details make Hodgin's work stronger and are a simple reminder to polish inside and out.

Over the past six years, Random International has incorporated the emergent properties of flocking behavior to light up art galleries around the world. Building custom hardware and meticulously crafting large 3-dimensional arrays of LEDs, Random International has effectively allowed these simulated living worlds to cross boundaries into our own architecture. Visitors witness the volumetric patterns from around the room;



Swarm Light, Random International (2010) however, the swarms pay no mind to their presence. The continued series of artworks engage viewers in a controlled space, and provide an experience that any passerby could engage with instantly. Random International plays with the legibility of the system as a part of their artwork, as they try to evoke the essence of a swarm, rather than the swarm itself.

Carpentry vs. Art

Ian Bogost, a well known game designer and critic, puts forth the idea of 'carpentry' in his book, Alien Phenomenology, suggesting that philosophers have always needed objects to think with. An astute Darius Kazemi points out in his blog[2], that this seems like this is simply the job of art, and so a distinction is made that 'carpentry' (not exclusive of art) is when matter is fashioned for philosophical use. In fact, Mens et Manus, latin for mind and hand, has long been the motto of MIT and AutomaTiles will quite literally leverage both the mind and the hand to become tools for thought.

The following section takes this natural step from artists that explore the subject of emergence through different media to artists, philosophers, and researchers that build tools to think with. Put simply, these are the objects out of which philosophy is made.

[2] http://tinysubversions.com/2012/04/notes-on-ian-bogosts-alien-phenomenology/

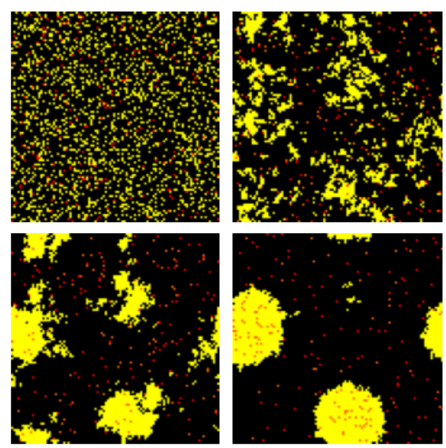
2.3. Precedents

Simple objects for playing with complex behavior are not particularly new and have been part of theses, art installations, and in some cases commercial products. The following section will cover an array of projects that are relevant and served to influence my work as well as create points of departure for how AutomaTiles carve their own space to investigate a new type of play and engagement with otherwise opaque systems. This area of devices is often called Distributed Computational Toys (DCTs), which is well documented in a brief survey of DCTs (Schweikardt). Many projects from the MIT Media Lab have explored similar form factors of giving digital systems tangible interfaces, such as SixFortybyFourEighty by Jamie Ziggelbaum and Marcelo Coehlo, Nami by Kelly Heaton, or Siftables by David Merril. Each of these projects explore different areas from thinking beyond the grid of the screen to the complexity of distributed networks.

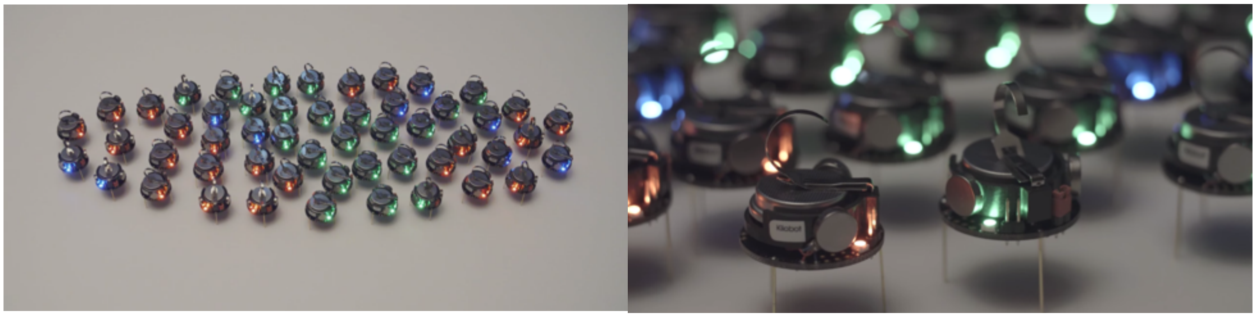
SixFortyByFourEighty places pixels in the physical world. (2011)

StarLogo

It is nearly impossible to discuss multi-agent systems for playful engagement and discovery without mentioning StarLogo, a software application for just that, designing simple rules for agents to explore emergent properties of a collective. In 1994, Mitchel Resnick expanded upon the child friendly Logo programming environment, where users would direct a single "turtle" to move in a given direction, to specialize in design for many "turtles" which can simulate many natural phenomena. Resnick details many of the successes as well as limitations of the platform in his book, *Turtles, Termites, and Traffic Jams*. Resnick is a master of story telling, each new ruleset connects seamlessly to real life experiences or playful fables. Understanding the value of narrative to instigate thinking about systems, eventually becomes the focus of AutomaTiles; without a clear path for storytelling, too many potential systems thinkers are excluded.



Simulation of termites in StarLogo emerging mounds of wood chips



Kilobots, 1024 autonomous robots from Wyss Institute help researchers discover and test mechanisms for self assembly. The above Kilobots are acting like pigment in a camouflaged fish, organizing into stripes with only local communication between robots. (2012)

Kilobots

At the Harvard University's Wyss Institute for Biologically Inspired Engineering, Radhika Nagpal runs the Self Organizing Systems Group, which is actively investigating how collections of individuals in nature cooperate to perform impressive tasks such as build shelter, find food, or even form a baby. Instead of simply simulating these systems in software, Nagpal has led the group to build a number of cooperative robotics, including Kilobots, a swarm of over 1000 robots. Kilobots move themselves and are not designed to be played with, but they act singularly, with no leader, and through clever algorithms, cooperate to accomplish a collective task.

The robots set a strong precedent for considerations around batteries and charging, ways to deploy code to a relatively large network of simple embedded devices, as well as the kinds of behaviors that might be fun to explore. Watching Kilobots evolve camouflaged striping patterns from random ordering is a delightful scene based off of Nagpal's algorithms (Nagpal). Even when simulating fireflies, the pattern seems random until every item synchronizes and flashes together. A simulation on a computer would handle this quite differently given the constraints of real world noise contribute to the experience.

BodaBlocks

In 2004, Leah Buechley, built a series of 3-dimensional cellular automata and a screen based interface for sending rules to the smart building blocks, dubbed BodaBlocks. Tiles could light up

BodaBlocks explore 3D cellular automata in physical space. (2004)

in two different colors, blue and green and used physical connectors placed between them to share signal. The blocks were designed for children to engage, play, and explore complex systems with the affordances of physical objects and cellular automata. It was valuable research to learn the best features, including the ability to pass rulesets from block to block as well as the hurdles of form factor and interface with a computer making the process quite cumbersome. The ambitious project successfully places toys for 3D cellular automata structures in kids' hands for a series of play tests.

Nami is a network of colorful orbs acting as individuals and behaving as a community. (1999)

SoundMites & Nami

A number of other projects related to distributed devices for playing with emergent behavior have come from the Media Lab, including SoundMites by David Bouchard and Nami by Kelly Heaton. While Heaton's Nami touches on classic artistic affordances like how one can paint given a networked array of "phyxels" (physical pixels), Bouchard works with a different sense, a soundscape produced by a field of devices. While the painting example is less abstract than most, both projects remain in abstract territory, leaving exploration and creation as their main purpose and function. The devices are not designed to be held, but can be handled; their scale suggests installation in a science museum rather than the form factor of a consumer product (Heaton, Bouchard).

SoundMites, a tool for exploring audio-visual emergent behavior. (2007)

Siftables/Sifteo

One of the first products most people think of when I mention tiles that talk to each other, is David Merril's Siftables, or later commercial product, Sifteo. Siftables were designed to explore the affordance of handling physical objects in digital contexts. Tiles respond to touch, tilt, neighboring tiles, as well as as communicate to a computer for use as a peripheral device. Each tile has a high resolution square screen for displaying full color images, at framerates of 15fps or higher. Since each of the tiles could display a multitude of information, the experience was built to work with 3 tiles or 12 tiles at maximum. While much of the hardware informs the possibilities that I am

Each Sifteo tile has a screen capable of displaying complex imagery. This application, like most applications designed for Sifteo, utilizes three tiles as a novel interface for interaction as well as a novel platform for games. (2011)

interested in exploring, the design suggests each tile is quite smart on its own, making the collective feel less special.

Each of the precedents described here engrained an image of the road already traveled and allowed me to focus on what would make AutomaTiles novel. Designing a friendly object is important; however, Sifteo's limited success shows that friendly hardware is not enough — the experience needs to guide developers to create rich content that engages users in a meaningful experience from the collective use of the tiles.

3. Approach

To accomplish my goal of building a platform for playing with simple and interconnected components which create complex behavior in their aggregate, I started designing from the ground up.

The hardware I have envisioned, designed and constructed differs from other similar devices due to its simplicity in components as well as interaction. I have used five principles in the design decisions made throughout:

- Feel in the hand, designed around the fingers
- Social in nature, seamless communication with neighbors
- Give them souls. Each device needs to feel alive
- Weigh the complexity of system interactions against simplicity of the interaction with any given tile
- Transparency. Most electronic devices are a black box, these are designed to show their face (a bit)

The AutomaTiles required the industrial design around the look, feel and behavior of each of the tiles, as well as design thinking for the kind of systems that could be simulated on them. The following section discusses my approach and the process of prototyping both hardware and software required for playing with complex systems.

3.1. Simulation

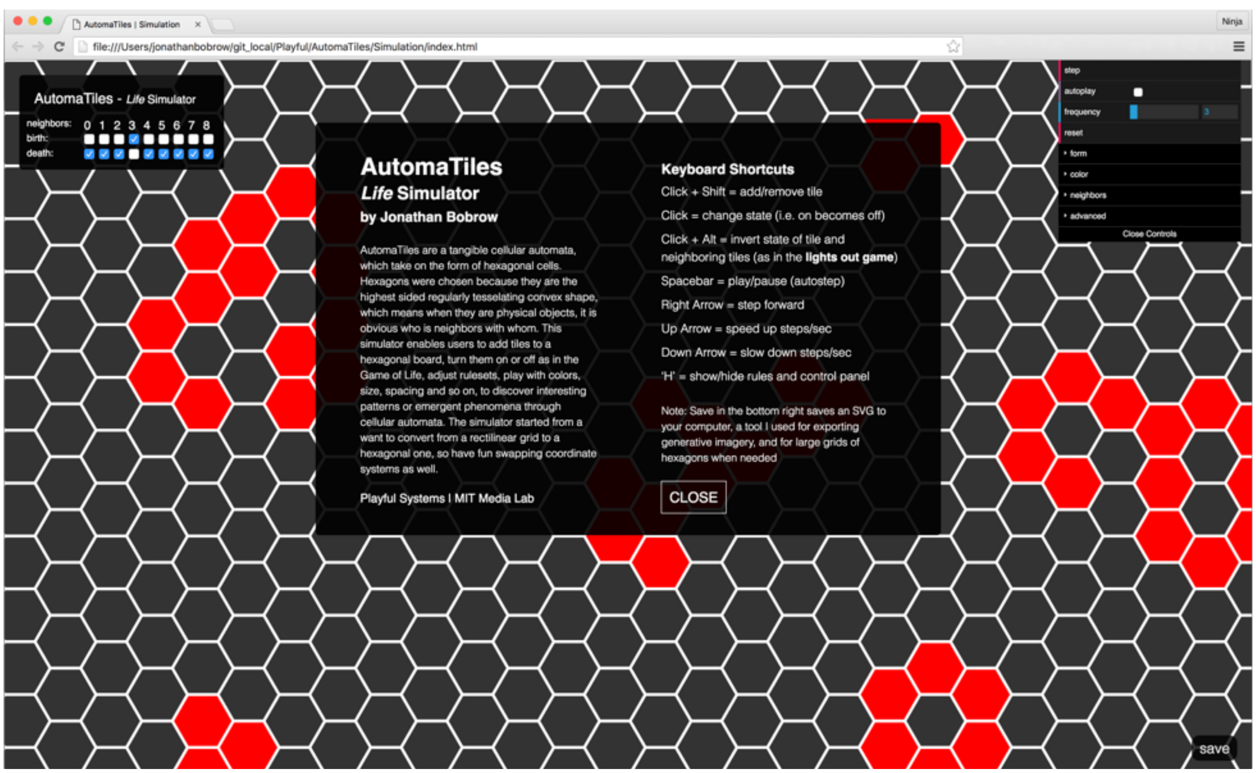
The design process for AutomaTiles began from sketches of the physical components, a very simple set of sensors and actuators to result in a much more complex and emergent behavior. The Media Lab has a funny way of making hardware appear as accessible and malleable as software, which isn't actually true anywhere in the world, not the Media Lab or even Shenzhen.

While I had a strong vision for the large design space afforded by the tiles, there were no such tiles in existence, and so in parallel, I built tools to simulate their behavior while Joshua Sloane (MIT '17, UROP) iterated on a robust design for the electronics inside them. To quickly build the simulator, I used javascript and made my platform in browser. The benefit of building for the web is being able to access the code anywhere, share it with anyone and gain feedback.

I will also explain in the chapter on games, how the simulator led to simulating the physical experience and later how I adapted an open source cellular automata simulator to prototype AutomaTile rulesets.

Creating tools to explore cellular automata

One of the best features about cellular automata is their inherent simplicity, most of them rely on a few lines of code. Developers often compete to see how quickly or elegantly they can arrive at an exciting or surprising ruleset. That means that most cellular automata visualizations are either quick sketches and applets (Java based web applications) or large robust



Here a version of the simulator is showing a full screen of AutomaTiles set to the Game of Life ruleset such that tiles become red if exactly 3 neighbors are red, and become grey if any more or any less are red. I have to assume that ruleset would become stable and not so interesting quickly, but it is easy enough to try quickly in the web simulator. The actual simulator is live at http://automatiles.com/sim. Note: SVG's animated and rendered w/ Two.js applications for discovering large scale solutions in Conway's Game of Life, like Golly.

I needed something more flexible than these tools so I could start to experiment both with the rulesets as well as the correct properties for the physical objects I was designing concurrently[3].

Finding the right shape

A rectangular form, like the pixel used for most cellular automata is great for the screen, but becomes problematic when thinking about neighboring tiles. In addition, rectangles suggest stacking rather than tiling, and immediately evoke computation and a rigid hierarchy. I thus chose to work with hexagons — the highest sided regularly tessellating convex shape — which immediately suggests tiling and all of those descriptors are important if I want to make an easily understandable object for the hand. These properties of hexagons explain why they come up frequently in system studies, from Donella Meadows to bees. Finally, removing the user from now-ingrained cartesian x-y rectilinear address systems is a way to defamiliarize and encourage exploration.

Logo (left) for Donella Meadow's Institute clarified for the not immediately obvious hexagonal shape represented in the design.

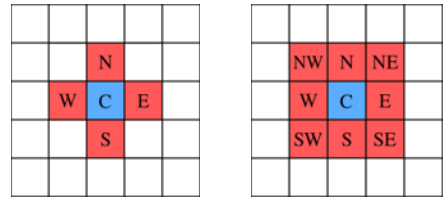
Games often use hexagonal boards to suggest a complex network such as territorial games or the graph paper that players use for Dungeons and Dragons. Hexagons suggest an accessible level of complexity and allow users to eschew the rigid and static affordances of rectangular blocks.

While a hexagon on screen feels arbitrary, the physical form lends it an essential feeling. The hexagon suggests a raw dynamic of communication between units. Communication between the standard rectangular cellular automata requires making a choice of neighborhoods.

The two types of neighborhoods described for square grid based cellular automata are Moore's neighborhood and Von

[3] The simulator I built used a 2D vector graphics renderer from Two.js and allowed for smooth transitions through both scale and shape. The simulator can be accessed and played with at http://automatiles.com/sim

Neumann's neighborhood. Moore's neighborhood contains the 8 surrounding squares, north, east, south, west, north-east, south-east, north-west, and south-west, which means that squares sharing an edge are neighbors and squares sharing a corner are neighbors, as well. This is the neighborhood model that Conway's Game of Life uses, as the high number of neighbors produces more potential outcomes for each interaction.



The two types of neighborhoods for cellular automata on a cartesian grid. The left diagram shows a Von Neumann neighborhood with only 4 neighbors in their cardinal directions. The right diagram shows a Moore neighborhood with 8 neighbors, showing diagonals included as neighboring cells.

In Von Neumann's neighborhood, a single cell only has four neighbors, north, south, east, west. These neighbors must share edges, which is more useful for a physical object, since edges are obvious signs of shared territory, much like a border for a state. The problem then becomes, how do we gain more neighbors while still sharing borders like the Von Neumann neighborhood. The solution there is to increase the amount of edges while making sure the shapes can be packed in a regularly tessellating grid.

The mathematical derivation[4] of the hexagon as the best solution for circle packing confirmed the intuition with which I began. Hexagons have indeed proved to be the ideal form for thinking about, engaging with, and playing with complex systems.

From Data Visualization to Data Tinkering

In 1971, Thomas Schelling described models of social systems evolving segregation[5]. He showed that small biases in large groups of people could result in unwanted segregation for the whole community. In 2014, Nicky Case and Vi Hart posted Parable of the Polygons[6], an interactive story based on these same models. Visitors to the site were told about two families, triangles and squares, and how each of them was only slightly "shapist." Triangles are happy around both triangles and squares, but prefer to be around more triangles than squares.

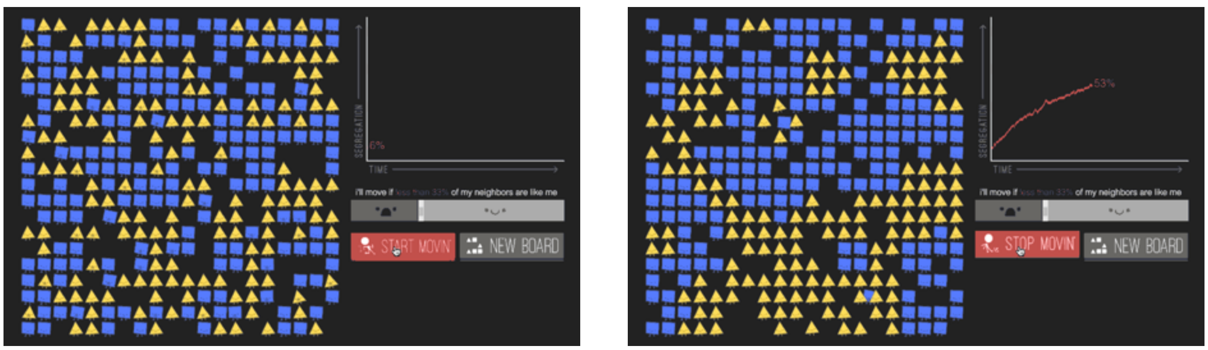
[4] https://arxiv.org/abs/1009.4322

[5] http://www.stat.berkeley.edu/~aldous/157/Papers/Schelling_Seg_Models.pdf

[6] http://ncase.me/polygons/

As the story progresses it becomes clear that this seemingly harmless act of wanting to be around more people like yourself, but not many more, results in great divides. The same information presented in a much more accessible and interactive way reached a very different audience and sparked new conversations about how unintended consequences can arise.

While this information is timeless, a number of injustices in the



Parable of the Polygons by Vi Hart and Nicky Case (2014) The left image shows a completely random distribution of triangles and squares and the right image shows how segregation evolves simply by wanting to move if less than 33% of your neighbors are like you.

United States of America involving police and their biased protection of citizens made this interactive even more impactful, upsetting, and timely. With Parable of the Polygons, people weren't simply told about segregation, they engaged in segregation through seemingly harmless actions they made themselves. The participation made this experience meaningful and allowed people to experience the complexity of social systems both first-hand and at a distance.

For AutomaTiles, the most valuable takeaway from Nicky and Vi's work is the aspect of storytelling, which makes this otherwise abstract system relatable. People were playing with data, but telling stories and seeing their own communities represented.

Abstract vs Representational Systems

The systems we most naturally understand represent something we are familiar with in this world. A study by Howard Gardner and colleagues at Harvard Graduate, made the distinction that there are people who like patterns and people who like to tell stories (of course there are people along that entire axis), and to reach the people who gravitate towards stories, AutomaTiles need to be a storytelling medium.

Red Reveal, a technique invented long ago, used here by designer Stefan Sagmeister to make compelling album art with juxtaposed encoded images.

Up until this point, AutomaTiles communicated to their users in one way, through abstract colors of light, on or off, or pulsating to signify the system underneath. While a design principle of AutomaTiles is to be simple and not use a high fidelity display, a solution for representing the systems visually came from a clever old trick. The embedded RGB LED can display a wide array of colors and if properly diffused, can flood the entire surface of the AutomaTile. Using the same technique as red reveal, seen demonstrated by cd album art by Stephan Sagmeister (see right), the light can be used to mask a single color at a time, resulting in two or even three unique images.

Series of 3 color prints reveal different images for each stage of a forrest fire using a technique akin to Red Reveal.

The first test was to simply display a happy and sad state for each tile, and so the happy face was printed in blue, only hiding when flooded with blue light, and the sad face was printed in red, only hiding when flooded with red light. The switching light switched the emotions on the tiles. With a simple printed icon of a smiley or frowning face, the system became legible to anyone that saw it. The next images I tried were designed to represent predator prey models of the insect world, and eventually I designed an image that could convey a fully grown tree, one currently on fire, or one that has already been burnt down. A fourth state of being struck by lightning could be added by simply flashing a bluish white light, since the flashing is already quite mimetic of lightning.

The above image shows the same printed skin for a forest fire AutomaTile over green light on the left, where the tree displays in full bloom, and over red-orange light on the right, hiding the leaves and revealing a tree set ablaze.

Interpretation of non-mimetic behavior

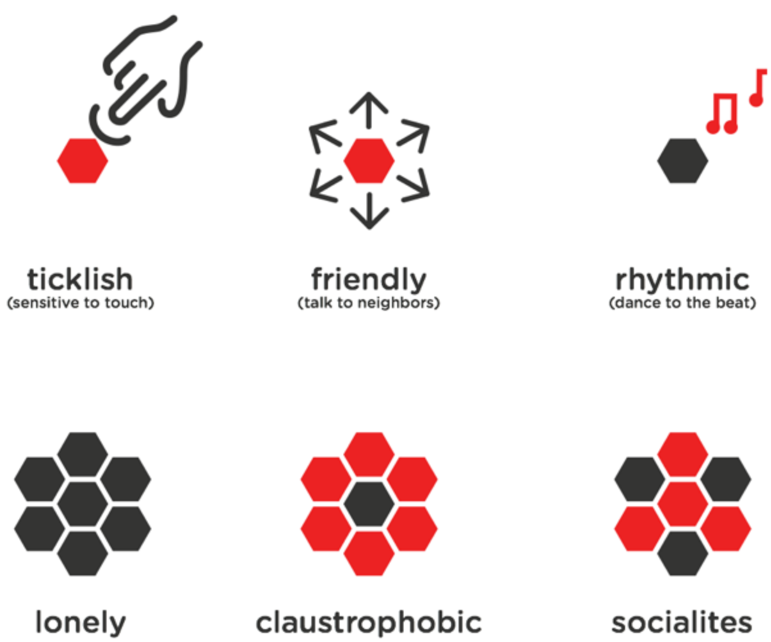
In researching prior DCTs, I have noticed that none have captured the minds of the general public to become a commercial success. I initially set out to capture the kind of complex systems and emergent behavior, however abstract, that excite me and friends alike, but learned over time that a large percentage of the population simply doesn't respond to the abstract patterns which pleases my mind as a sort of poetry.

In game studies, Henry Jenkins was the first to articulate "narratological" vs "ludological" approaches— in short, understanding the world in stories or systems. This is obviously not binary and exclusive, but points to the tension between the two. AutomaTiles are fundamentally "ludological" but I have set out to understand and develop "narratological" layers to "tap the emotional residue of previous narrative experiences" (Jenkins 2).

Adding an expressive face to the red hexagon changes the color to a meaningful blush.

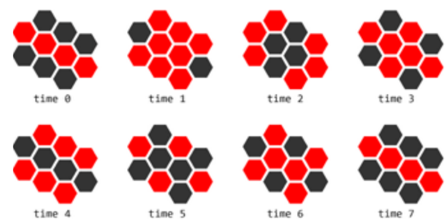
In Conway's Game of Life, for example, from simple non-mimetic patterns, we implicitly classify patterns and behaviors as lifelike. This is an instinctive human action to make sense of the world and in developing AutomaTiles, I have set out to understand (and even guide) what those interpretations might be. Providing a vocabulary is necessary to guiding interpretation and use. This emerges naturally through use; some of these emerged in the process of testing and explaining the various patterns with users.

Once I gave the AutomaTiles some character, it was easier to assign and describe what they do. Being sensitive to touch meant an AutomaTile was ticklish. The ability to communicate with neighbors makes them friendly (even though in reality, their communications could be quite confrontational).



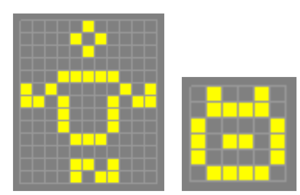
Lastly, their microphone or ability to listen (greatly limited to simple peaks) makes the tiles rhythmic.

Depending on the rules that the tiles are told to follow, the next properties could change drastically. In the life-like ruleset, it made sense to link a single tile's happiness or sadness to its loneliness, claustrophobia, or desire to party (participate in dynamic equilibrium). These small local rulesets might be witnessed when playing with the system, but most likely these three simple behaviors of a single tile result in a surprising emergence from the entire population. In fact, the life-like ruleset can spread happiness quickly and find a number stable and dynamic equilibria(calm country side or a bustling city) depending on the arrangement.



AutomaTiles: With 10 tiles, a pair of swing dancers strut their stuff. The dance is 7 steps long and repeats over and over again.

Some examples of using this vocabulary can be seen when describing some of the simplest forms in the AutomaTiles Life-like ruleset. Starting with only three tiles, we can refer to all three tiles in a happy state sharing two neighbors each a "rock." The three tiles will remain happy and so the end result is a solid form, that is easy to imagine resembling a rock. Four tiles is more interesting, with a form resembling a "humming bird", or seven tiles revealing a "windmill." More tiles create forms that look like amulets, beating hearts, swing dancers, or an active game of duck-duck-goose[7]. Conway's Game of Life remains a common reference because of the deep lexicon with contributions from computer scientists, authors, and hobbyists.



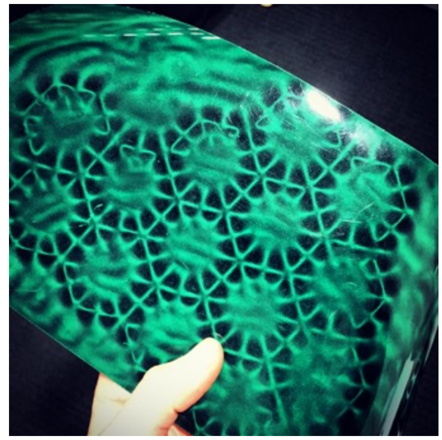
Conway's Game of Life: The left cells are in a formation called the cauldron and the right is a Cheshire Cat.

The section on games will go into more detail about the properties of this life-like ruleset, but the important takeaway for the approach is how these properties of AutomaTiles get translated to the hardware that will need to not only perform each of these functions, but evoke these functions to their user so that their purpose is legible. If each AutomaTile looks and feels friendly, or is believably ticklish, and hints to its rhythmic properties, the form's behavior becomes more accessible through narratological means.

[7] A series of these figures are shown as gifs here: http://automatiles.tumblr.com/post/129871033941/simulation-in-parallel-to-the-design-and-technical and can also be simulated here: http://automatiles.com/sim

4. Hardware

AutomaTiles are unapologetically tangible. From the very first sketch and ideation of AutomaTiles, I was clear that the physicality would be the central feature, and a desirable tangible interaction is what would make this experience unique. All of the research-based distributed computational toys (DCTs) don't fully consider the sensuous qualities of their handling. The tiles went through iterations of materials as well as form factors. Standing on the shoulders of board game designers such as Hive[8], I was able to quickly approximate weights and sizes for pieces that would be comfortable to hold, and desirable to play with.

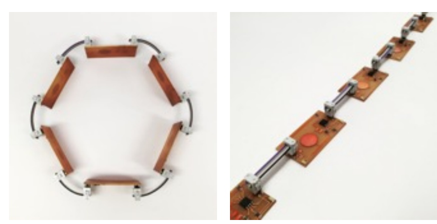


Holding magnetic film to the bottom of AutomaTiles reveals a beautiful pattern of connectivity. The magnets make connections in the same way the tiles communicate, a symmetric handshake on all six sides.

4.1. Prototyping

The journey for AutomaTiles has been lengthy, and they started in a very different place than they ended up. To quickly explore the physical nature of cellular automata as a novice with electronics, I decided to prototype a version of the AutomaTiles with only a single dimension of communication. Each tile could communicate to its neighboring tile to its left and to its right, just like a Wolfram cellular automata[9]. Stephen Wolfram's *A New Kind of Science* suggests even this amount of simplicity can produce interesting and complex emergent patterns.

The first AutomaTiles were realized and aesthetically pleasing, which is the first thing that everyone noticed, but were limited in the ways one might interact with them. They could be connected and rearranged, but the objects felt as though they failed to enter a dialogue, as a tool for thought should. Through their form they would evolve to feel natural to interact with.



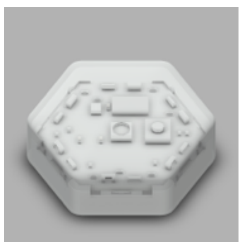
Holding magnetic film to the bottom of AutomaTiles reveals a beautiful pattern of connectivity. The magnets make connections in the same way the tiles communicate, a symmetric handshake on all six sides.

[8] Hive is an award winning hexagonal board game with smooth "bakelite" tiles. http://www.gen42.com/hive

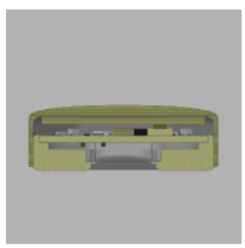
[9] Stephen Wolfram explored one-dimensional cellular automata (CA) exhaustively which filled many of the pages of *A New Kind of Science*. The depth allowed from a single strand of cellular automata suggested that starting with a single dimension would not limit my complexity, however, I would suggest that it limits the systems legibility. Note that most singular dimension CA's are displayed on a two-dimensional grid, with the y-axis serving as a timeline. Seeing the pattern over time makes the CA instantly legible rather than cryptic flashing and blinking.

4.2. Form Follows Function…

Or the other way around. As a designer, I always felt that the relationship between form and function is a matter of perspective. The AutomaTile's form is informed by its function, but its function should be communicated through its form. When developing both the function and the form together, there is a feedback loop articulated by Jony Ive, Apple's Chief Design Officer, where hardware informs the software and software informs the hardware in harmonious unison. Apple credits much of its success to its closed system with the aforementioned end to end control over every product. Unlike Apple, however, AutomaTiles are designed to be an open system, in every sense, causing a natural tension with meticulous design and craft by a single person. My notes on the design and development of the form and function follow.





Final Form

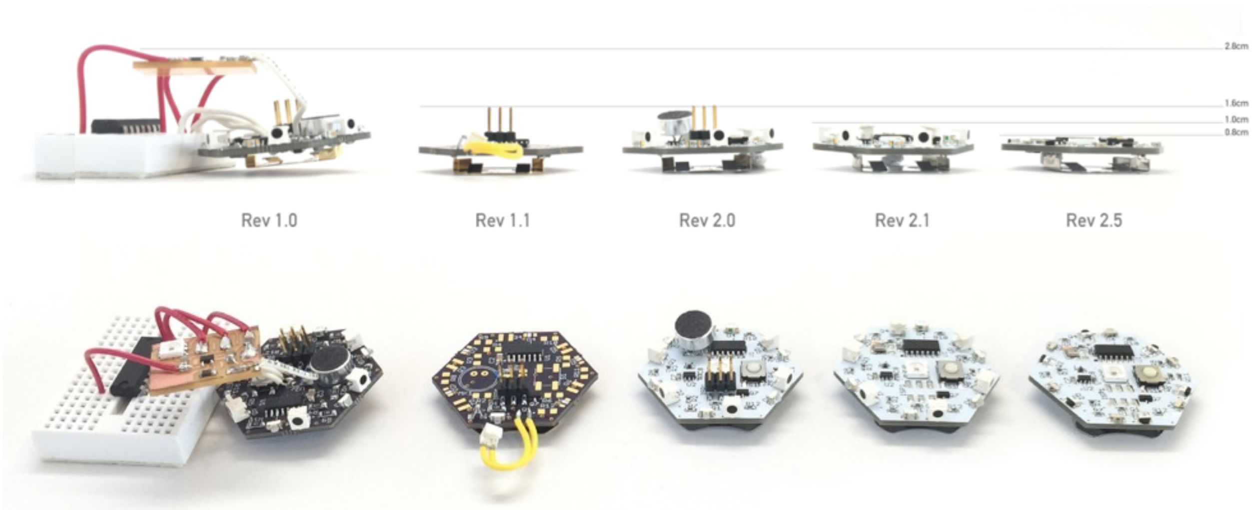
AutomaTiles are not a product, so the final form is a public beta, which has been successful in communicating to users their affordances while remaining friendly to hold. This in turn has allowed me to gain valuable feedback for the many iterations.

The enclosures did not have a path as direct as the electronics, since material tradeoffs are subjective. The entire AutomaTile consists of 3 parts:

- the Printed Circuit Board (which will be discussed shortly in the electronics section)

- the Base which holds the PCB and magnets for aligning with neighbors
- the Top Enclosure which is comprised of a soft membrane and a rigid surface for use as a physical button as well as light diffusion

All of the components together weigh a carefully measured 22 grams per tile informed by the Hive tiles. Each tile feels like a solid unit rather than a loosely assembled electronic device. I prototyped many different materials, weights, and mechanisms for each bottom and top enclosure to perform in the hand as effectively as the electronics perform inside the unit.



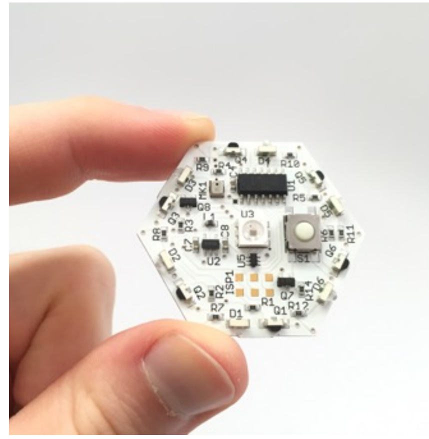
4.3. Electronics

The electronic requirements for AutomaTiles are relatively straightforward. Each of the precedent setting DCTs approached many of the design constraints I have set for AutomaTiles, such as battery power, need for communication protocol and often times, relatively small construction. The circuit board's size was defined by the fit in the hand. Looking to standards for handled objects[10], the average child's hand is ~4cm across and 8cm long which felt agreeable with the maximum dimension of the very first hexagonal AutomaTile PCBs measuring ~4cm as well.

[10] http://www.technologystudent.com/joints/edu8.htm

The Revision Process

Prototyping electronics follows a similar mantra as working with wood, which is “measure twice, cut once.” The very first AutomaTiles had simple error such as a backwards component or an incorrect voltage for control, which required the equivalent of duck tape for a circuit board. Each board had patch wires and in some cases, copper tape to run traces carefully from the incorrect pad, to a new location. By the third revision of the electronics, each trace was measured three times over, since we were making 100 of them, and every subsystem was first verified in an electronic mockup, either breadboarded or made with a quick turn PCB.



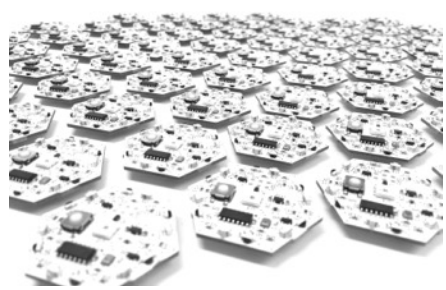
Top side of AutomaTile Rev. 3

AutomaTiles require the following inputs and outputs:

- 6 digital input pins for phototransistors (listen to neighbors)
- 1 digital output pin for IR LEDS (broadcasting to neighbors)
- 1 analog to digital input for Microphone (detect snaps)
- 2 digital output pins for RGB LED (visual display of color)
- 1 digital input for momentary button (respond to touch)
- 1 digital output pin for power switching (while in sleep mode)

Bottom side of AutomaTile Rev. 3
board design by: Joshua Sloane
assembled by PCB.NG

The ATtiny84 has 11 digital I/O pins, and AutomaTiles require 12, so the button and the IR LEDs share one pin, meaning that for the moment a user is pressing an AutomaTile, it momentarily pauses broadcasting to its neighbors.



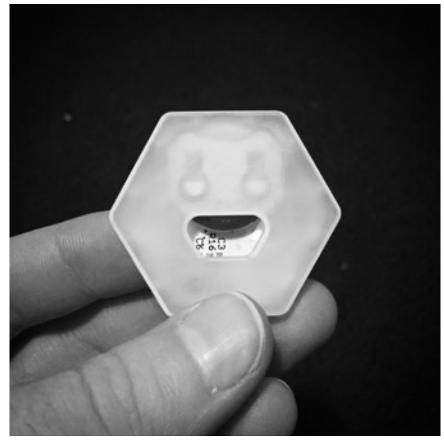
100 assembled PCBS, AutomaTile Rev. 3

Batteries

Electronics always beg two questions, how is it powered, and how long will it last? In this case, I have decided to make the first run of AutomaTiles run on single use Lithium coin cell batteries that will eventually require replacement. To those ends, careful consideration was placed in designing for the replacement of the battery, and even signaling the ability to do so (see early prototype at right).

Each AutomaTile has a variety of running systems: the LED, transmission, reception, microphone, and microprocessor.

These systems can be thought of as the eyes, ears, nose, mouth, and brain of each AutomaTile. Each of them uses active energy while they're in use, and some of them require passive energy[11] even when asleep. To minimize energy draw, I added a transistor to play the role of a power strip for the unnecessary systems while asleep[12]. AutomaTiles should be easy to use and feel like living objects, that so long as they are powered—they "sleep" like humans, rather than turning off like a machine.



The bottom of the case was designed to allow for easy access to the battery, and resulted in a bit of paradolia.

In conclusion, it appears that AutomaTiles will get 45 hours of active use and over 600 hours of sleep in the worst conditions[13] as detailed above. This should provide at least a month of very active use and potentially much longer depending on use. In commercial settings I think a half year of use would be the goal.

In seeking to optimize for the lowest power consumption, we measured each system while active (on), inactive(off), and asleep (low power mode). The amount of time spent in each of these states was then used to calculate an effective current rate. This information combined with the battery capacity at two bounding rates provided an accurate estimation of total play time as well as storage time. The chart below details these values.

System	**Current On(mA)**	**Current Off(mA)**	**Current Sleep(mA)**	**% On (time in use)**	**Effective Current**
Microcontroller	1	0	0.001	100%	1.00
LED	2	1	0	50%	1.50
IR LEDs	2	0	0	8.33%	0.17
Phototransistors	0.4	0.3	0.3	8.33%	0.31
Microphone	0.8	0	0	100%	0.80
Total	**6.2**	**1.3**	**0.301**		**3.78**

Additionally, with the price of rechargeable batteries coming down, that could be an option, as it is attractive to consider both ecologically and for convenience.

---

[11] Systems of the electronics that were consuming lots of energy while asleep on revision 2 were the voltage booster, which is converting the 3v power source to 5v for powering the RGB LED. The voltage booster can simply have its supply cut of by a MOSFET, making sure that it is only consuming energy when in an awake state. Additionally, the infrared phototransistors are sensing whether or not a neighbor has woken up, and any infrared light that hits those components consumes a little bit of energy from the battery. Note that the 3rd revision will run the battery down if left in the presence of IR light, such as the sun.

[12] The sleep routine is much like our own bodies, which regulate our energy use by lying down, closing our eyes, and reducing the number of systems actively online. Speaking mimetically about the AutomaTiles results in mimetic features.

[13] Battery lifetime is calculated by using the battery life curves presented in the datasheets. Battery lives are shortened when electronics are pulling a lot of current, and AutomaTiles live on the edge in this regard. At 0.3mA, the battery life is a healthy 225mAh, at 4.0mA, the battery life is a less appealing 150mAh. The chart shows the effective current draw based on how often each subsystem is active when AutomaTiles are awake.

4.4. Firmware

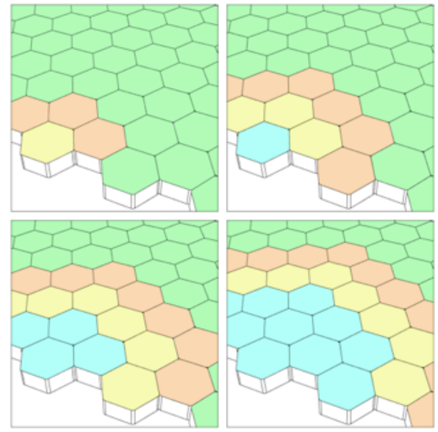
Using an ATtiny84 as the brain of the device limits the kind of functions the AutomaTiles can perform, but I see this as a feature. The complexity is not supposed to be injected into a single tile, but rather be revealed through their interactions. Half of the original 8Kb of flash storage available on each tile is used by the firmware through pin definitions, interrupt routines, and all of the ADC logic, among others. This leaves about 4Kb for a maker's program to be uploaded, which is more than enough for all use cases we have tested so far. We have noticed that verbose programming is discouraged by this limited space, but this only seems to be a problem when we have attempted to add complex behavior to a single tile, which is counter to the ethos of AutomaTiles.

Starting from the moment they are awoken, AutomaTiles share their liveliness by waking their neighbors. This process could happen in milliseconds, seemingly instantly; however, the speed

AutomaTile wake sequence is initiated by a single woken AutomaTile (by touching it), the neighboring tiles are then woken and signal their neighbors and so on until the entire population has woken from their slumber. The above tiles show yellow (appears white in photo) to signal waking up, and then show their state, displayed red here, from before they went to sleep. State persisting through sleep is optional but on by default.

at which they wake each other is governed to a rate that feels legible. Pressing on a single AutomaTile to wake it and seeing everyone respond immediately is much less believable than seeing a wave of lights turning on, as though word has spread through the community that play time is here.

Even more compelling than their wake sequence is their ability to gossip or share parameters of rulesets. Similar to the wake sequence, the user is signaled that a tile is entering a program state by pulsing green, transmitting is yellow, and receiving is orange, and completion of the process finishes with blue before returning to a previous state. The cascade of colors disperses much like an oddly cautious game of telephone, confirming receipt along the way.



Transmission of parameters through a dense lattice of AutomaTiles. (a wave of gossiping AutomaTiles)

4.5. API

All of the AutomaTile code is written in C, and is optimized to save space on the tiny processor. Most novice or hobby electronic tinkerers are familiar with Arduino for being the most easily accessible and hassle free solution for electronics tinkering. More and more, Arduino hardware is used in public installations and shockingly robust engineering and the IDE, while minimal, just works.

For convenience to users, the codebase was refactored to be a hardware type in the Arduino IDE. By selecting AutomaTile from the board dropdown, all of the functions associated with AutomaTiles become available. Getting neighbor states is as easy as calling `getNeighborStates()`, which will return an array of the 6 neighbor states sensed. The API has been used for a number of examples, many of which will be explored in the following section on the AutomaTiles games.

Providing a familiar interface for programming the AutomaTiles is an important aspect of making them accessible on every level. The API is currently being expanded to allow for gossip of particular parameters from tile to tile. Incorporating distributed methods into the framework only helps to reinforce that AutomaTiles are a tool for systems thinking.

Please see Appendix B for detailed documentation of the API or visit the documentation on Github[14].

[14] Documentation of the AutomaTiles API: https://github.com/jbobrow/AutomaTiles/blob/master/Examples/API.md

5. Games

Introducing a new object into a foreign environment requires careful consideration, and when inventing a new platform, it needs to show its colors early on, so people will know what to expect and how to expect to interact with it. The form, electronics and API are all part of this consideration, but the way AutomaTiles behave once powered is what defines the platform. Where AutomaTiles really shine is through the games that can be developed with and for them. The following section follows my process of creating games and finding the novel spaces afforded by this new platform.

Prior Experience

My experience in making games has always been in the design of games with extrinsic goals. I have made games to increase the speed with which soldiers identify anomalies, teach officers soft skills for conflict mediation, or introduce secondary school children to the many ways we can impact our energy consumption.

For each of these games, there was already a system whose parameters needed adjusting, and the conflict resided in having parameters in competition with each other. This is not the way all game designers work, or how all games come about, but often there is a system that presents challenges for the player, and the fun arises from that conflict.

For AutomaTiles, my design process works from the small actions that produce a system, and require modification or reorganization of that system to win. This “bottom-up” approach is very different than thinking about a system as a whole and then trying to emulate the parts. Often times the kind of properties that emerge from a simple ruleset given to the tiles will be surprising to the designer and developer working on the game. Just like other game design processes, only through iterative design and play testing is it possible to arrive at a well balanced and fun game.

Game Making

The following discusses a number of experiments or games built and played with on AutomaTiles and will break down the affordances of AutomaTiles to eventually define a set of best practices for the platform.

Game of Life inspired ruleset for a play test of a life-like game.

Choosing a ruleset and then coming up with a fun mechanic is some of the fun of working with AutomaTiles. Using the API I developed, the designer can rapidly prototype and iterate the game either as rules (the conventions that players use in the game, e.g., taking turns) or code (the way the system handles those rules).

Games can be made out of literally any kind of system, designers just need to balance them in such a way that players feel engaged or challenged and that the signal to noise[15] ratio is high. Keeping players involved in a game means that each person feels they can contribute or have agency in the outcome of a game. A player's agency is dependent on the legibility of the system, thus designing legible systems for AutomaTiles becomes paramount.

First steps: Game of Life

The very first ruleset I explored was a ruleset based on John Conway's Game of Life. While the rules are very simple, in aggregate, it is very difficult for someone to grasp how a certain pattern emerges or fails to emerge, and so a game around these rules need to take that into consideration. The rules for the AutomaTiles embodiment of the Game of Life were this, if a tile is touching 2 or 3 blue tiles, the next step forward it time, it will be blue, and if a tile is touching any more or any less blue tiles, it will be red. And instead of calling them blue or red, blue represents happy, and red represents sad, otherwise known as alive and dead, but it is easier to explain why tiles

---

[15] The term "signal to noise" come up frequently when talking with Kevin Slavin, as it was common to discuss projects in this way at New York University Interactive Telecommunications Program(NYU ITP), where a project might not be communicating anything to its viewer. If the information trying to be conveyed could simply be replaced with noise, then the experience, visual or otherwise, is not doing its job. It is worth noting that in the case where noise is the point, this is a less applicable rubric.

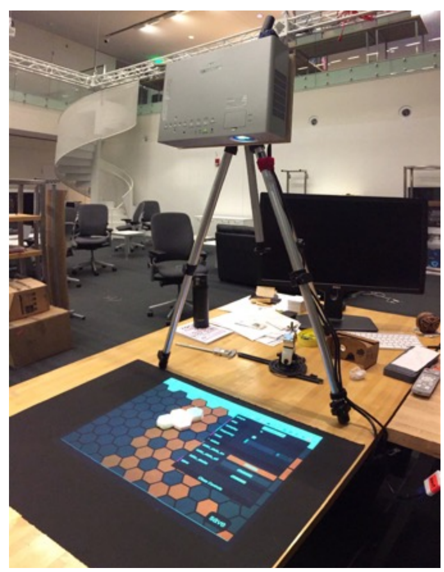
become happy again, a littler more difficult to explain why they resurrect from the dead.

With these rules, there are static equilibria and dynamic equilibria that will always be reached. This means that if you step forward in time to infinity, the tiles will have created a pattern that repeats itself, much like chasing lights, or that the tiles will be static, and not change at all. The repeating pattern can be given a period of repetition and a static form is simply a repeating pattern with a period of one time step. It is also possible to differentiate between a static state where blue tiles remain on the board, versus an all red board, which I consider to be a dead board, since there is no potential energy in the system or possibility for life.

The first game I tried with these tiles was a competitive game where each player places a tile in either a blue or red state, and then first player to make the board reach a dynamic equilibrium wins. Similarly, the game can be played where the first person to reach a static equilibrium that is not a dead board wins.

To bring the game to life before the hardware was finished, I used a projector with my browser based simulator, which I built to easily projection map physical tiles for handling. This technique is a simplified version of the kind of augmented reality from Fluid Interfaces (i.e. LuminAR).

I led a number of play tests with these rules and found that there were a number of aspects that made the game a difficult game to play for 2 players. The first bit of friction was that the system was difficult to understand. This would be acceptable if gameplay makes the system more legible, but that was not our experience. The second difficult aspect is that we needed to let time go to infinity or at least until an equilibrium was seen after each turn. When the board is small, this happens quickly, but as the board grows bigger, it is seemingly unpredictable. After placing more constraints, and trying a few different iterations of this life-like game, it didn't seem to be a promising road



Using a projector and hexagonal 3D printed tiles to create an Augmented Reality mockup of AutomaTiles.

forward. I concluded that this particular game is not a good match for this ruleset.

Lifegrams

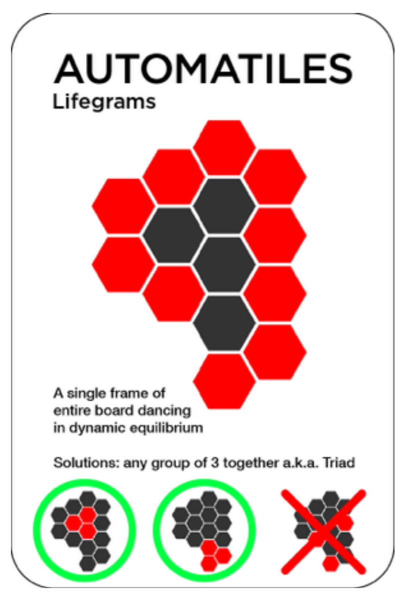
Continuing with the life-like ruleset, I moved to the space of a solitary game, where the single player has a deck of cards, with goals to arrive at, just like in the game Tangrams. One version of this game simply has an arrangement of tiles, that the player sets up, and then the card asks the user to set an initial condition that will result in the correct equilibrium. For example, starting with only three blue tiles, arrive at this static equilibrium with all blue tiles around the edge.

The answer is not obvious how to get there, and there might be more than one way to do it (always true on a symmetric board), but the player is challenged to learn how the system evolves. In this example, the player is asked to reach dynamic equilibrium with every tile, only starting with three of them happy(blue). I observed that players quickly learn that placing all three happy tiles together no matter where they are in this configuration is the only way to spread happiness so quickly across the entire board. It doesn't matter where they start as long as they are together.

Front side of a Lifegram card.

Another card might ask to start with a largely happy population in an initial condition such that it results in a small part of the population in a happy equilibrium. The combination of form building and description of the wanted behavior asks the player to engage with the emergent properties and learn about some common patterns or techniques. The next step with this game would be to play test this ruleset with a deck of cards, thoughtfully prepared, and see how engaging the gameplay is, as well as what is afforded by the physicality that a simulated version of the game doesn't.

For further development, I set out to start with the physicality of the tiles and use those affordances to guide the game design process.



Back side of a Lifegram card.

Forest Fires

There is a whole series of games that can be developed from the idea of outbreak and containment, in fact, Cordon Sanitaire, a Playful Systems game relies on this mechanism to see how coordination emerges among total strangers without communication. The mechanic of trying to contain some unstoppable force is fun because of the complexity that arises from the various permutable possibilities. Similarly, being responsible for the spread can be fun since the complexity and often rapid growth or decay that emerges from simply seeding a contagion is rewarding. With AutomaTiles, a couple of models fit this dynamic, and the first to be explored was a common complex systems example, a forest fire.

A forest fire, (terribly simplified), can be simulated by having a single tile represent a plot of land. That plot of land can represent fertile soil, a tree, or a tree on fire. If a tree is struck by

Sparse layout, forest fire ruleset. Trees will grow slowly when not surrounded by other trees, but forest fires will travel slowly and be easy to contain.

Dense layout, forest fire ruleset. Trees will grow quickly, surrounded by other trees, however forest fires will spread quickly, clearing the fertile land.

lightning, it catches fire (if fertile soil is struck by lightning, it remains fertile). If a tree is next to a tree or many trees on fire, then this tree will catch fire too. Eventually the tree will provide no more fuel for the fire, and it will return to being fertile soil. This cycle repeats and the process of forests burning and growing back is simulated in a quick loop (sometimes too quick). This ruleset makes for a nice demonstration of how one tile affects its neighbors, just like gossip or some other phenomenon, and it also lends itself to gameplay.

To first test this ruleset, I adopted a tool being built and open sourced by Nicky Case[16]. I modified the tool to support hexagonal neighborhoods on both the back-end and front-end. To give new rulesets to each of the tiles, the tool uses "natural" language to describe how each of them behave based on probabilistic or deterministic language. In my experience, this makes for quick adjustments to a specific type of rulesets, for which forest fires were the poster child (a screenshot to the right). From this simulation, it was an easy transition to using the API and programming the actual AutomaTiles to play with the forest fires hands on. Forward thinking, Nicky suggested building a tool to simply export from this application to AutomaTiles, which would be a nice way to lower the barrier of entry to programming and customizing rulesets on the tiles.

A modified version of Nicky Case's Simulating the World in Emoji. http://automatiles.com/sim/emoji

At the MIT Play Day, young children were first given the forest fire tiles and asked to arrange them to grow trees the fastest. Each user would group the tiles as compact as they could, soon revealing a full forest with bright green trees. I would then press down on one of their trees and strike lightning, causing a fire to spread through their dense forest. They were then challenged to find a way to protect their forest by rearranging the tiles. Some of the children quickly separated all of the tiles, making it nearly impossible for a fire to spread, but also growth was quite slow. Other children placed the trees in padded rows, much like a farm we would recognize today. Each of these solutions had their advantages and disadvantages, and these tradeoffs lead to game play.

The game I envision for forest fires can exist as a single player game, in which Mother Nature decides when the storms might strike. A player can do their best to keep their forest alive despite the inclement weather. Another version of the same

[16] Simulating the World in Emoji is an explorable explanation at http://ncase.me/simulating/

game has two players, each planting trees of their own species. Imagine one player plants pine trees, represented with blue-green light, and another player plants birch trees, represented by yellow-green light, and each move a player can either build their territory or attempt to harm their neighbors. The two players must always build on one continuous board, so a fire could have dire consequences for both players, or should I say, “backfire.”

Rules for Backfire (A game about forest fires)

Setup: For 2 players. Each player gets half of the tiles and sets the tiles to be different types of trees (holding a tile down switches its type).

Goal: Grow your trees into a full forrest or out survive your opponent.

Turns: To make a move, players place a single fertile tile. Players must place tiles adding to the same graph (one connected board). Each turn is a step forward in time. Each time step trees may grow and fire may spread (if there is fire on the board).

There is a small chance that lightning might strike each time step and with more tiles on the board, the chances go up. Probability for tree growth is based on density of fertile land, so trees grow faster in denser areas, but fires also spread quicker through denser areas. Players must balance racing towards planting a full forest and laying out a fireproofed garden.

Fracture board setup. All 4 players segregated but joined in the middle.

Fracture move mechanism, split or *fracture* the board into two pieces and join them in any way, as long as it fits, to complete your turn.

Fracture: A Game about Diversity

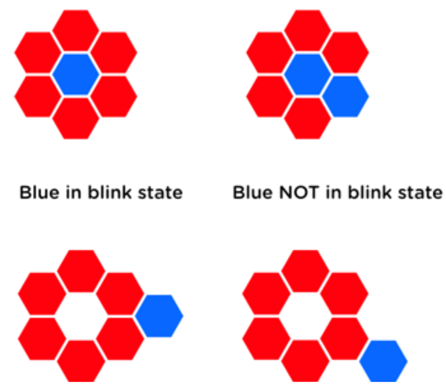
While hosting Celia Pierce, Associate Professor of Game Design at Northeastern University, and her cohort of game designers[17], they had an idea for a cooperative game with the goal of making a board happier through diversity. The game rules are simple, a color is happiest when it is around the most diverse group of colors and sad otherwise.

The algorithm for this is straightforward, something achievable by someone with very modest programming skills and so it was a matter of 15 minutes before we had our code written using the AutomaTiles API. Brightness of the AutomaTiles was mapped linearly to how diverse its surroundings are, so a more diverse area will shine brighter. We added a constraint based on loneliness, such that no tile is happy unless it has at least two neighboring tiles. To help with legibility of the system, we decided that a tile with no similar neighbors will blink to show its maximum excited state.

---

[17] Fracture game development by: Celia Pearce, Jeanie Choi, Isabella Carlson, Mike Lazer-Walker, Joshua Sloane, and Jonathan Bobrow as well as play testing help from Miguel Perez and Benjamin Berman.

We began testing the game, seeing how we could best work together to make the board diverse. From this process we arrived at a few great conclusions. While I knew the tiles were difficult to move from the middle of a board (due to their form), I hadn't thought of an alternative for modifying the middle of the board. This is when Mike Lazer-Walker suggested splitting the board into two halves, or fracturing the board to make a move. The satisfaction of splitting the magnetically attached tiles made the game more fun and sparked a social interaction that board games thrive on. The fracture mechanism requires just enough dexterity that some moves are more difficult to execute than others[18].



One version of the game suggests cooperative play and any number of players could contribute to solve a sort of map coloring problem, such that no single color neighbors the same color. Another version of the game that immediately appealed to the group prototyping was a competitive one, where each player is a color, joined into a segregated starting block, and players take turns to try and make their tiles diverse first. This is the game that we lovingly call Fracture.

Rules for Fracture

Setup: For 3-6 players. Each player gets 5-6 tiles of a specific color and places their single color tiles together. Players then join their colors in the middle, and the game starts with all tiles connected, but completely segregated by color.

Goal: Get your tiles to be touching only other players tiles (different colors than your own) and touching at least two tiles (to avoid being lonely).

---

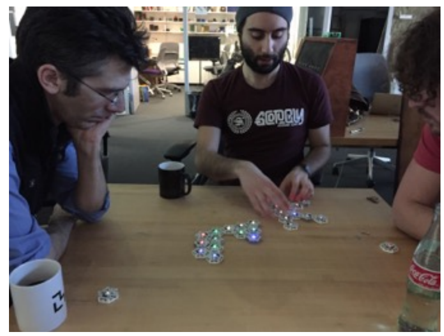
[18] It could be worth investigating if dexterity limits strategic play, but the dexterity required is so minimal, it isn't apparent that it would limit a player greatly or so much as to challenge competitive play.

On a similar note, when designing any object for mass consumption, it is important to consider the accessibility of an object. AutomaTiles, while tactile, only communicate through light, limiting the audience for engagement. Looking forward, it will be an exciting and interesting task to expand the senses AutomaTiles can use to communicate.

Turns: To make a move, players fracture the board into two separate pieces(no more) and then rearrange the pieces to be put back together in a new formation. Players take turns in a clockwise rotation.

When a single tile is placed touching at least two other tiles and neighboring only other color tiles, it will flash with excitement. The first player to have all of their tiles flash is the winner.

The gameplay initially felt like it might simply go around in circles with players preventing each other from winning, but our first game resulted in an unexpected winner. We were driven to play again to see if we could find more strategy than was visible from the surface when we first played.



Playful Systems group plays a round of Fracture. Here Kevin, a first time player, is watching how other players strategize as well as feel the geometry for their move.

The game relies on managing your own drive to win as well as preventing other players from winning, especially the person who moves directly after you. The rest of the bookkeeping is done by the board itself. While Fracture could in theory be played with a printed set of cardboard pieces, the amount of bookkeeping to play an engaging game relies on the intelligence in the board to let players always be aware of game state. While video games have been able to communicate the state of a game through flashy animations and highlighting areas of danger, board games have never shared this affordance. This is one of the best features of a game made with AutomaTiles, that the board can take care of the bookkeeping and allow players to engage in games of higher complexity than previously possible for board games.

Just because the board is showing state doesn't mean that there aren't tiles strategically placed to be freed of their neighbors all in one swift move. The possibility space for moves is enormous, but learnability is instant. This sets a strong precedent for how AutomaTile games should feel, both in the mechanic for moving as well as ease of use and ability to master the game.

Prototyping Ruleset

To aid in the process of developing a platform, we hosted sessions with game designers to understand the ways people want to use AutomaTiles. Prototyping consisted of idea generation around game genres, interaction methods, as well as generating games for systems already simulated on the AutomaTiles.

One such session led to a wonderful finding by Mike Lazer-Walker, that simply being able to use the tiles as smart facsimiles for imagined game play would allow the designer to quickly demonstrate or play test before having to put any algorithms on the devices themselves. This can be looked at as the equivalent of paper prototyping for AutomaTiles games.

The two immediate needs were a range of colors that can be quickly changed on the tile without needing programming, and switching between a blinking state and a solid state. Even though AutomaTiles can conceivably display millions of colors, it is fair to assume that only a small fraction will be discernible as different by the general public. A good place to start for unique colors (although this changes based on local[19]), is ROYGBIV. The prototyping ruleset has seven colors: red, orange, yellow, green, cyan, blue, magenta, and they are all accessibly by simply pressing for more than 1 second on a single tile to get to the next color or double tapping in less than a second to arrive at the previous color. For blink modes, simply pressing quickly on a tile will change it from solid to a slow blink pattern, then a fast blink pattern, and back to solid. On this prototyping ruleset, the interaction between neighbors is imagined and can later be added when the designer has come up with an interesting game idea or mechanic.

Some game design comes from playing with an existing system, but this prototyping ruleset allows for the designer to only be guided by the physical affordances of AutomaTiles and their

---

19 A study by Berlin and Kay to determine the constraints on cross cultural color naming. In short people draw the line in different places for what the colors of a rainbow might be (since it is a continuous spectrum, the divisions are in some sense arbitrary) http://www1.icsi.berkeley.edu/wcs/

modes of interaction. Having a play test for gameplay or a hands on design session with people not familiar with the platform highlights areas that might be overlooked, and allows us to reprioritize aspects of the platform. This particular play test resulted in a quick to make prototyping tool that will surely help game designers in the next session.

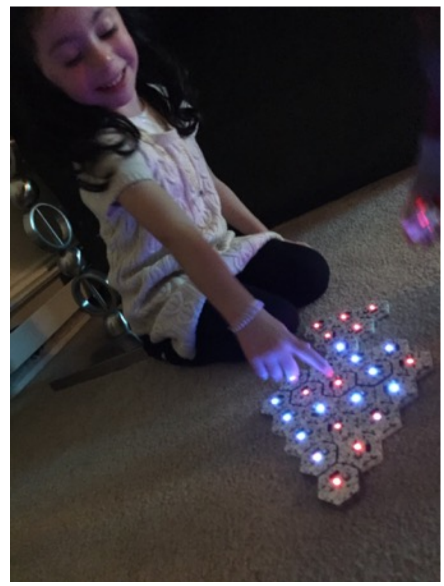
Best Practices

Developing a new platform requires developing a language to design for the platform, and as with any language, it will be better for saying some things over others. If you try hard enough, you can express nearly anything with a language, but this section is an exercise in constraints.

AutomaTiles have a simple set of components, and each of those components affords possible interactions. For example, the microphone is simple a way for global communication. Every single tile hears the same thing (and if they don't they let their neighbors know for redundancy). One obvious use for a global notification was to step forward time, but the microphone could also be selective in the future to respond differently to different pitches enabling an interesting hybrid tabletop rhythm game. In my investigation, the other components lend themselves to specific use cases as well, and their uses do not need to be limited by the initial intent of the hardware design.

Each Tile responds to touch in a number of ways. The intent of allowing touch was initially to seed information into the system. Just as a cellular automata needs to be seeded in an initial state, the button was intended to setup these conditions; however, in practice, the button could serve far more interesting purposes. In the case of the forest fire, the button could be used for striking lightning, which is hardly an initial condition, and through discussions with game designers, the idea of revealing information through touch became a recurring topic. Instead of putting information into the system, the touch of a tile could reveal something about the system. In a traditional board game, this could be looked at as a face down card, or a pile of facedown cards, but the difference here is that these face down cards can change according to their neighbors. This novel use let the aggregate of the tiles know something the players don't, and only through player interaction, the truth is revealed.

To support the idea of hidden information, the tiles RGB LED doesn't need to be directly linked to the tile's state. In this case, a tile could be dark until polled to reveal identity. Designers even suggested using a similar tactic to the red reveal of the forest fire skins to provide partial information to players. In this case, players could wear filtered lenses and see different versions of the same board, with hidden information as well as shared information. Expanding the use of a single light's ability to communicate through color was a delightful surprise from a limiting feature of the tiles themselves.



My four year old cousin playing with AutomaTiles simply as a toy. What she saw as a living christmas tree, her brother saw as a spaceship. The patterns were exciting at times, but hardly legible.

Each side of the AutomaTile broadcasts information to neighbors as well as listens to neighboring tiles, and this is done through a quick burst of IR light. Since the tiles have skins that can be placed over the tiles and changed for different games or scenarios, the skins could be designed to effectively block communication for certain sides to expand the possibilities of directional tiles.

This was particularly exciting for use as an alternative to Northeastern University's electronic quilting game, eBee (Pearce). Blocking communication for some of the tiles would allow the tiles to serve the purpose of "routing electricity" from one base to another and completing a circuit.
Lastly, the solid construction of the AutomaTiles and their strong magnets afford gameplay inspired by other dynamic tabletop games. Sliding the tiles into each other makes for a more strategic version of shuffleboard, not dissimilar to the popular arcade game Puzzle Bobble[20]. Stacking is also possible, and suggests a weird spin on Jenga[21] with rules around the dynamic relationship of the tiles.

---

[20] Also known as Bust-A-Move https://en.wikipedia.org/wiki/Puzzle_Bobble

[21] Jenga is a stacking tower game with small wooden pieces whose imperfections allow the tower to be disassembled and assembled higher, piece by piece http://www.jenga.com/

Future Game Exploration

Each of these inputs or outputs, when handed over to developers for creative use afford many more possibilities than originally imagined. As long as there is a story to tell for why a button behaves a certain way, or a reason for neighbors not communicating, it helps build a narrative, the gateway for thinking about systems. The future games can certainly expand on these affordances to build rich narratorial environments.

In the course of running workshops for AutomaTile game design, friends and guests ideated and began to design components for games to be played on the new platform. Here is a list of potential game types generated at the Game Maker's Guild brainstorming session:

- Asymmetric Strategy - one player tries to make all tiles blink, the other player tries to make all tiles red
- Capture the field - control areas of the board, much like go
- Othello + stuff - add special tiles to a classic capture game
- Risk Style game - With a real time component from the tiles
- Query the board - i.e. ask the board if it is in the correct arrangement, kind of like mastermind
- Cooperative Puzzle - solved puzzle shows win state
- Competitive Puzzle - timed gameplay
- Musical Application - being on key to unlock the next stage

(full notes covered in Appendix D)

Conclusion

Looking forward to the possibilities afforded by AutomaTiles, the three directions — toy, game, and tool — each have interesting and exciting possibilities ahead. The use of each term provides a lens and a framework for the user's experience. As a *toy*, AutomaTiles become an instrument for exploration and imagination. G*ames* provide goals and engage users in challenging and rewarding tasks. As a *tool*, the tiles provide a new way of interacting with complex systems.

Toy

Through user testing, as toys, AutomaTiles have clearly conveyed that they are easy and desirable to handle, with a low barrier of entry for play. Watching user interactions made clear that AutomaTiles also immediately communicate to users that they are responding to their neighbors, afforded by the feedback of the system. It is worth noting that some systems are much easier to grasp than others, so starting with a simple ruleset with obvious behavior, lends itself to quicker understanding.

In Mitchel Resnick's *Turtles, Termites, and Traffic Jams*, he concludes that people are comfortable thinking in centralized terms (120). People assume that a flock has a leader, or that someone or something is singularly responsible for the patterns that arise. Resnick witnesses this tendency even while his subjects were exposed to StarLogo, a playground for decentralized thinking. I set out to get a sense of whether this applied to AutomaTiles.

When I brought AutomaTiles to the MIT Museum Play Day, part of the Cambridge Science Festival, children from the age of four and up played with the tiles simulating forest fires as well as rules for diversity. All of the children interviewed responded with comments of interaction between the tiles, and when asked if they had control over the system, even the four year old responded, "somewhat," alluding to the properties inherent to the system. When asked what was happening or to describe what they were seeing, the term "evolve" was common or the use of fire "spreading" rather than moving. I cannot say conclusively that working with StarLogo on a screen versus playing with AutomaTiles as separate units in the hand is the reason for this distinction, since the world is a different place with even more distributed context than was around for the time of Resnick's publishing. However, my hypothesis is that the intangible, interchangeable and fungible nature of pixels implicitly suggests these kinds of dynamic behaviors. To be able to model and experience them in the physical world seemed to be, for the children I evaluated, novel.

While children did get to use the AutomaTiles as toys, it was with limited time and not measured with quantifiable or scientific rigor. Continued work and investigation with the platform will bring about more evidence as to the value of the toys. For use as a toy, and tool for exploration of decentralized thinking, there were promising comments; children did exclaim that the tiles were a delightful experience. Moreover, it was clear from the inquiries for purchase from both children and parents, that AutomaTiles would be welcome in any home.

Tool

There has been interest in AutomaTiles as a tool from museums, including the Liberty Science Center in New Jersey, individuals from New England Complex Systems Institute (NECSI), Martin Nowak's Program for Evolutionary Dynamics, and a variety of fields dealing with complex behavior have all mentioned their interest in using the tiles to help others understand their respective studies of systems.

From the perspective of tool development, it is certainly more efficient to explore these systems on a screen. The affordances of physical interaction, however, become more important when individuals can work together in the most familiar forms of collaboration we have: *face-to-face real-time physical interaction*. While a computer's GPU is optimized for cellular automata simulations with great efficiencies in speed, the value afforded in AutomaTiles is in providing real-time collaborative perspective.

As a tool, AutomaTiles should feel like "carpentry" (Bogost). Just as our bodies need space to explore and build muscle through trial and error, our minds, too, can use objects, or collection of objects, to exercise and grow stronger. It was therefore necessary to make the objects feel welcome in the hand; a feature noted by everyone who holds them.

Professor Hiroshi Ishii, a global thought leader for tangible interfaces, would refer to AutomaTiles as a Graspable User Interface, encouraging manipulation with both hands (or in the case of multiple users, many hands) and resulting in parallel computation(Ishii, 3). While modern touch based interfaces such as the iPad allow two handed control, the hands are often limited to a reduced 2 dimensional plane for for input. Screen based tools for interaction with dynamic systems are limited in their bandwidth for communication and therefore "fall short of embracing the richness of human senses and skills people have developed through a lifetime of interaction with the physical world" (Ishii, 7). This affordance of physicality, among others, makes objects-to-think-with a new type of tool for a new kind of thinking.

> **"I do not think at all when cubing. I can talk freely about anything while doing world class speed. It seems like I just need to hook up my eyes and hands to some independent processor part of my brain, and it will just solve it before my eyes"**
>
> *- Lars Petrus, World Class Rubik's Cuber*[22]

Games

There is something special that happens with games that require handling, related to muscle memory, or as my colleague puts it, the Cerebellum. Apparently, this area of the brain is responsible for taking care of motor skills and does so without

---

[22] The competitive world of Rubik's cube solving is a global sport and Petrus not only held records as early at 1982, but also documented and shared his strategies for competing , but also held thttp://lar5.com/cube/menthol.html

interfering with other forms of thought. While Fracture has not been played with long enough by a single person or group of people to develop a deep knowledge of the play space and strategy, it is conceivable that players could develop a deeper understanding of the game through both their eyes and their hands, much like a Chess Grandmaster, Othello Champion, or Rubik's Cube competitor.

The possibility for the mind to work in parallel for gameplay speaks to a more holistic form of gameplay. After playing baseball for 20+ years, I can engage aspects of the game that no beginner would have the mental capacity to manage. So much of the game has been offloaded to another part of my brain, that I can deeply contemplate the intricacies of the game while I watch or play. For most games and most people the physical games they play are limited in their complexity because the rules and "bookkeeping" have to be held in their brain. But by providing people with physical interactions around emergent behaviors or complex systems, AutomaTiles introduce a unique way for tabletop games to maintain the synchronous, social aspects of games and incorporate computational complexity.

The ancient game of Go has such an enormous possibility space that it had recently been considered unsolvable. Were a computer to iterate through all of the possibilities, it would last many lifetimes. People describe how Go players approach the game — they 'feel' the board, since it's so arduous to 'compute' the possible branching paths, even for a computer[23] — so players are in fact processing complex branching systems, but not in a purely procedural manner. They are "thinking with their hands."

This is something that games are good at exposing, these ways we engage with systems in ways that are not explicitly procedural in our own metacognitive consideration. Games

[23] Recurrent Neural Networks are a technique for using probabilistic models for computation. Inspired by current intuition for how human brains process complex data, an RNN relies on multiple layers of abstraction to then arrive at a result with a fidelity proportionate to its given duration.

developed for AutomaTiles both afford a large possibility space, but can communicate to a user some of the properties of that possibility space. For example, Fracture tiles signal the players about the distribution of the current boards game state, something a professional Go player takes years to acquire.

Lastly, games is an extensive category, and I have described in detail some of the ways that Automatiles change the types of games we play, and how we play them. There's no name yet for the emerging field in which games take on the synchronous face-to-face interaction of board games and the complex computation of video games. But I hope that Automatiles adds to the vocabulary that the field can draw from.

When I began this thesis I set out to create a platform for playful engagement with complex systems. As Automatiles matured and expanded, the work led me down a non-linear path, with children, complex systems theorists, and board game designers. Whether as a tool, a toy, or even a space for storytelling, there is much left to explore. In the meantime, I have contributed an open source platform[24] extending across hardware, firmware and software, ready for future investigation by me and by anyone with similar goals.

Edith Ackermann articulated the ways that artifacts serve as a tool for dialogue and "do so by opening up greater or lesser mental elbow room (*Spielraum* in German)" (Ackermann, 3). This *Spielraum* should be rich and playful, and it should be enticing enough to shape the worldview of a generation as it prepares to deal with the complexity that lies ahead. My own journey with Automatiles led me to understand that no real system is simple and every system is a matter of balance. I hope that someday it can lead to a similar path for its users.

[24] https://github.com/jbobrow/AutomaTiles

Bibliography

Ackermann, Edith. EXPERIENCES OF ARTIFACTS: People's Appropriations / Objects' 'Affordances.' 2007. Web. http://web.media.mit.edu/~edith/publications/2007.explo.ofarti.pdf

Buechley, Leah, Eisenberg, Mike, 2007. Boda Blocks: a collaborative tool for tangible three-dimensional cellular automata. In: Proceedings of CSCL'07, pp. 102–104.

Bogost, Ian. Alien Phenomenology, Or, What It's like to Be a Thing. Minneapolis: U of Minnesota, 2012. Print.

Bouchard, David. (2007). Embodied Emergence: Distributed Computing Manipulatives. (Masters Thesis). Retrieved from https://dspace.mit.edu/handle/1721.1/41743

Gardner, Martin. "MATHEMATICAL GAMES The fantastic combinations of John Conway's new solitaire game *life*." Scientific American 223, October 1970: 120-123. Print.
Retrieved from http://www.ibiblio.org/lifepatterns/october1970.html

Heaton, Kelly. (2000). Physical Pixels. (Masters Thesis).
Retrieved from https://dspace.mit.edu/handle/1721.1/9298

Ishii, Hiroshi, and Brygg Ullmer. "Tangible Bits." Proceedings of the SIGCHI Conference on Human Factors in Computing Systems - CHI '97 (1997): n. pag. Web.

Jenkins, Henry. Game Design as Narrative Architecture, in First Person. New
Media as Story, Performance, and Game (eds.) Pat Harrigan and Noah Wardrip-Fruin.
MIT Press, Cambridge 2003 (in press).

Johnson, Steven. Emergence: The Connected Lives of Ants, Brains, Cities, and Software. New York: Scribner, 2001. Print.

Meadows, Donella H., and Diana Wright. Thinking in Systems: A Primer. White River Junction, VT: Chelsea Green Pub., 2008. Print.

Merrill, David, Jeevan Kalanithi, and Pattie Maes. "Siftables." Proceedings of the 1st International Conference on Tangible and Embedded Interaction - TEI '07 (2007): n. pag. Web.

Nagpal, Radhika. "Engineering Amorphous Systems, Using Global-to-Local Compilation." Understanding Complex Systems Complex Engineered Systems (n.d.): 291-306. Web.

Pearce, Celia and Carlsson, Isabella et al. eBee: An Electronics Quilting Bee and Game. 2015. Retrieved from http://sokath.com/main/files/1/smith-siggraphstudio15.pdf

Poundstone, William, and Robert T. Wainwright. The Recursive Universe: Cosmic Complexity and the Limits of Scientific Knowledge. New York: Morrow, 1985. Print.

Resnick, Mitchel. Turtles, Termites, and Traffic Jams: Explorations in Massively Parallel Microworlds. Cambridge, MA: MIT, 1994. Print.

Reynolds, C. W. (1987) Flocks, Herds, and Schools: A Distributed Behavioral Model, in Computer Graphics, 21(4) (SIGGRAPH '87 Conference Proceedings) pages 25-34.

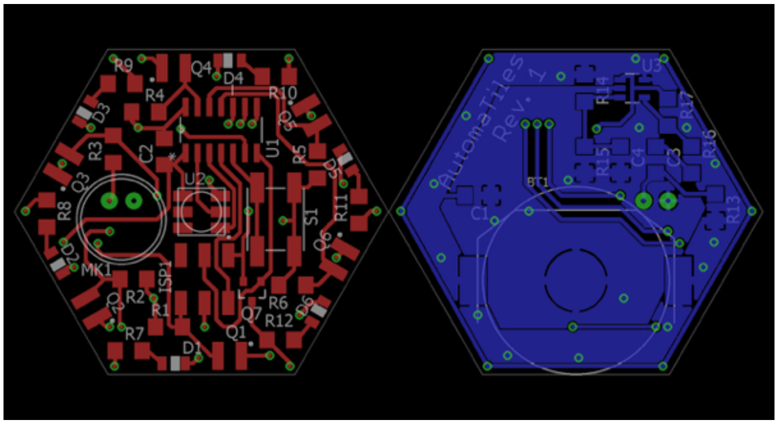
Schweikardt, Eric, and Mark D. Gross. "A Brief Survey of Distributed Computational Toys." 2007 First IEEE International Workshop on Digital Game and Intelligent Toy Enhanced Learning (DIGITEL'07) (2007): n. pag. Web.

Wolfram, Stephen. A New Kind of Science. Champaign, IL: Wolfram Media, 2002. Print.

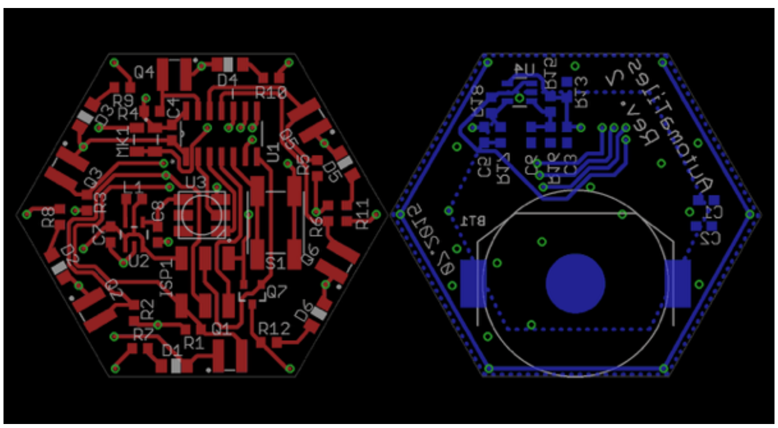
Yamins, Daniel and Nagpal, Radhika. Automated global-to-local programming in 1-d spatial multi-agent systems. pages 615--622, 2008.

Appendix A

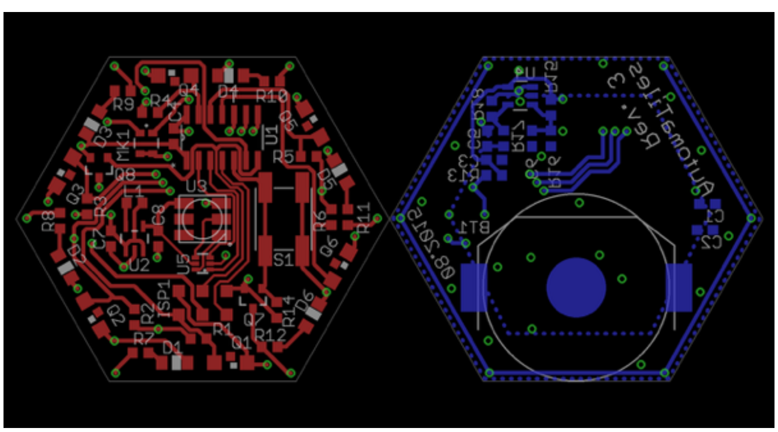
Circuit board design for the 3 revisions of AutomaTiles PCBs.



Revision 1, 06.2016 (reversed IR phototransistor, low voltage to LED)



Revision 2, 07.2016 (reversed MEMS mic footprint)



Revision 3, 08.2016 (current revision)

Appendix B

The following is the GitHub repository associated with AutomaTiles. Each firmware version history is contained at http://github.com/jbobrow/AutomaTiles

AutomaTiles API

Since the API is quite short, this is the current documentation for using it. It should be apparent on how to use it from some of the examples, but this can serve as a nice reference while you are writing your first programs for the AutomaTiles.

Blank sketch

```
void setup() {
  setButtonCallback(button);
  setStepCallback(step);
}

void loop() {
  // continuous time logic here
}

void button() {
  // handle button down here
}

void step() {
  // discrete time logic here
}
```

AutomaTile methods

getNeighborStates

```
void getNeighborStates(uint8_t * result);
// pass it an array for the size of the neighborhood and it will return each neighb or's state in that array.
```

sendStep

```
void sendStep(void);
// step forward in time, simulate ruleset
// maximum frequency 10Hz
// these do not cue up, i.e. if you send step more frequently, steps will be dropped
```

getTimer

```
uint32_t getTimer(void);
// returns a value of time in milliseconds (starts at 0 from battery in)
```

setColor

```
void setColor(const uint8_t color[3]);
// send this function (R,G,B) 0-255, color will change on next cycle
```

setState

```
void setState(uint8_t state);
// set the state of self 0-15 (limited due to communication of state and step frequency)
// invalid states are ignored, i.e. return without state change
// NOTE: empty spaces act as tiles of state 0 (treat this accordingly)
```

getState

```
uint8_t getState(void);
// get's our own current state, 0-15
```

setStepCallback

```
void setStepCallback(cb_func cb);
// pass your own function to setStepCallback, this function takes no arguments and returns
nothing
// can be set once in the beginning of
d during runtime
// defaults to do nothing (change this
```

setButtonCallback

```
void setButtonCallback(cb_func cb);
// pass your own function to setButtonCallback, this function takes no arguments and returns
nothing
// can be set once in the beginning of your application, but can also be hot-swapped during
runtime
// defaults to do nothing (change this to default to flash blue)
// NOTE: you cannot delay in the button callback (i.e. timer will not update while you are in
the callback)
```

setLongButtonCallback

```
void setLongButtonCallback(cb_func cb, uint16_t ms);
// pass your own function to setLongButtonCallback, this function takes no argument s and
returns nothing
// long button callback will be activated after a button has been pressed for n milliseconds
(second parameter)
// can be set once in the beginning of your application, but can also be hot-swapped during
runtime
// defaults to do nothing (change this to default to flash blue)
```

```
// NOTE: you cannot delay in the button callback (i.e. timer will not update while your application, but can also be hot-swapped to default to flash red... or white) you are in the callback)
```

setLongButtonCallbackTime

```
void setLongButtonCallbackTime(uint16_t ms);
// set the amount of time it takes before the longButtonCallback is triggered // can be hot-swapped during runtime
```

setTimerCallback

```
void setTimerCallback(cb_func cb, uint16_t ms);
// set a custom timed callback, whatever you want, it will fire after a given amount of time
// can be hot-swapped during runtime
```

setTimerCallbackTime

```
void setTimerCallbackTime(uint16_t ms);
// set the time for your callback separately (maybe you want to update that time after setting it initially, perhaps it evolves over time)
// can be hot-swapped during runtime
```

setTimeout

```
void setTimeout(uint8_t seconds);
// sets the amount of time before sleeping in seconds
// defaults to 20 seconds
// if you want the automatile to never sleep send it 0, this will run down your battery
```

setMicOn

```
void setMicOn(void);
// turns the microphone on (i.e. listens for snaps)
```

setMicOff

```
void setMicOff(void);
// turns the microphone off (i.e. only steps forward from neighbor notification)
```

Appendix C

The following are notes generated from Game Makers Guild game designers. The workshop was held on April 25, 2016, for an approximately 2 hour session including an introduction to AutomaTiles, the origins as well as the API, a quick overview of best practices, and a subsequent free design period. All designers presented their ideas at the end of the design session and the following are notes generated from the session.

Assymmetric Strategy Games

2 player
Randomized board where
1 player has a goal related to blinkiness
the other has a goal related to color/colorwheels

2-N player
Competive puzzle solving
-random goal card (colors, shape, and/or blinkiness)
-each player w/same set of tiles
-race to finish (like Bananagrams)

Boat Navigation Through minefield
boat has set # of sonar pulses to detect mines. Has to get through.

Voice Trainer
light indicates the note to sing changes for new note. Could make a sing-a-long where a string lights up as you get the notes
NAME THAT TUNE! Guessing game based off of

Mastermind
1 player has set behind a screen.
Other players working to replicate.

Competitive Mastermind
Set up row of tiles as your puzzle.
Setup row of tiles as your guess.
Press buttons to get answers about the accuracy of your guess.

Maze
Use the covers & a figurine. Figurine takes a step by clicking the next tiles button. Gets a sonar reading of surrounding

Space Ship Wars - shoot other ships' hexes off.

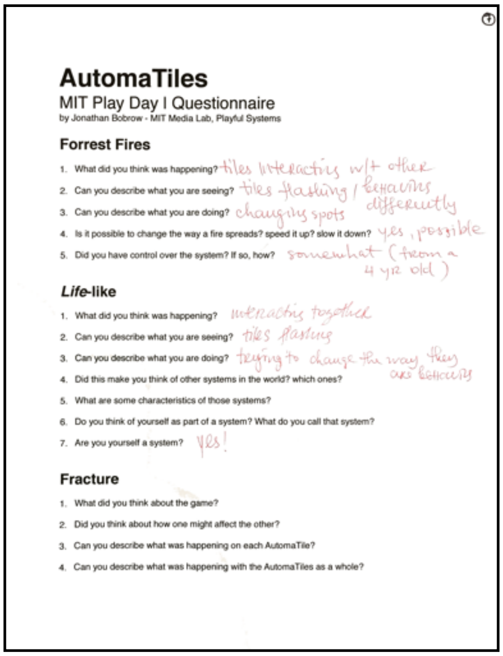
Figure out how the tiles behave to solve a puzzle (make all purple, but rules change).

IDEAS FOR GAMES

- MAKING MY OWN GAME "CARNIVORES" MORE INTUITIVE
- IMPLEMENTING THE iOS GAME "SYMMETRY" IN MEATSPACE
- BE THE FIRST PLAYER TO EXCITE A GLOBAL STATE THAT DOESN'T DECAY TO A STATE STEADY STATE IN A GIVEN NUMBER OF STEPS
- GAME WHERE PLAYERS MUST INFER THE RULES WHILE THEY PLAY
- DECKBUILDING
- INDUCTIVE LOGIC GAME WHERE GOAL IS TO BE 1ST TO INFER THE "LAW OF NATURE" THAT DETERMINES WHETHER A PARTICULAR PATTERN FLASHES OR NOT. (SEE THE TABLE GAME "ZENDO")

Appendix D

A sample of my questionnaire from MIT Museum Play Day. The open play session was held on April 20, 2016, as part of the Cambridge Science Festival. AutomaTiles were on display and out for play for hours with a variety of rulesets, Fracture, Forest Fire[25], and Life-like. Children from the age of four years old to adults had the chance to engage with AutomaTiles and tell me what they thought.



AutomaTiles

MIT Play Day I Questionnaire

by Jonathan Bobrow - MIT Media Lab, Playful Systems

Forrest Fires

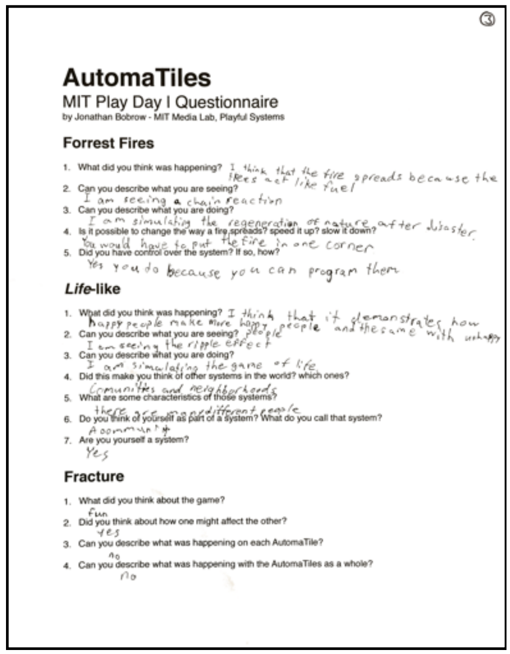
1. What did you think was happening? tiles interacting w/+ other
2. Can you describe what you are seeing? tiles flashing / behaving differently
3. Can you describe what you are doing? changing spots
4. Is it possible to change the way a fire spreads? speed it up? slow it down? yes, possible
5. Did you have control over the system? If so, how? somewhat (from a 4 yr old)

***Life*-like**

1. What did you think was happening? interacting together
2. Can you describe what you are seeing? tiles flashing
3. Can you describe what you are doing? trying to change the way they are behaving
4. Did this make you think of other systems in the world? which ones?
5. What are some characteristics of those systems?
6. Do you think of yourself as part of a system? What do you call that system?
7. Are you yourself a system? yes!

Fracture

1. What did you think about the game?
2. Did you think about how one might affect the other?
3. Can you describe what was happening on each AutomaTile?
4. Can you describe what was happening with the AutomaTiles as a whole?



AutomaTiles

MIT Play Day I Questionnaire

by Jonathan Bobrow - MIT Media Lab, Playful Systems

Forrest Fires

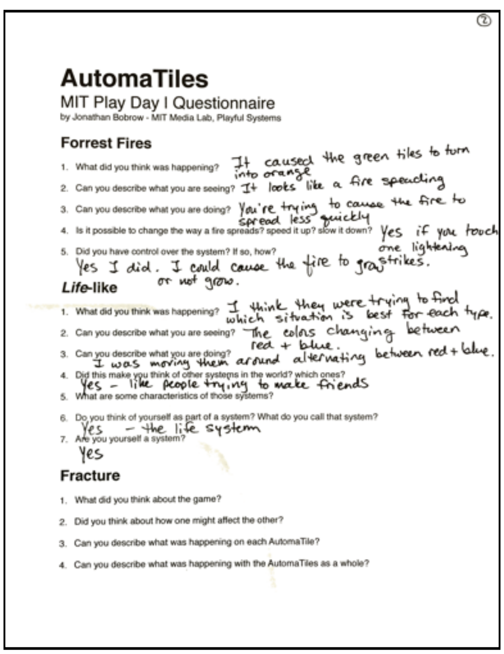
1. What did you think was happening? I think that the fire spreads because the trees act like fuel
2. Can you describe what you are seeing? I am seeing a chain reaction
3. Can you describe what you are doing? I am simulating the regeneration of nature after disaster
4. Is it possible to change the way a fire spreads? speed it up? slow it down? You would have to put the fire in one corner
5. Did you have control over the system? If so, how? Yes you do because you can program them

***Life*-like**

1. What did you think was happening? I think that it demonstrates how happy people make more happy people and the same with unhappy people
2. Can you describe what you are seeing? I am seeing the ripple effect
3. Can you describe what you are doing? I am simulating the game of life
4. Did this make you think of other systems in the world? which ones? Comunities and neighborhoods
5. What are some characteristics of those systems? there are more different people
6. Do you think of yourself as part of a system? What do you call that system? A community
7. Are you yourself a system? Yes

Fracture

1. What did you think about the game? fun
2. Did you think about how one might affect the other? yes
3. Can you describe what was happening on each AutomaTile? no
4. Can you describe what was happening with the AutomaTiles as a whole? no



AutomaTiles

MIT Play Day I Questionnaire

by Jonathan Bobrow - MIT Media Lab, Playful Systems

Forrest Fires

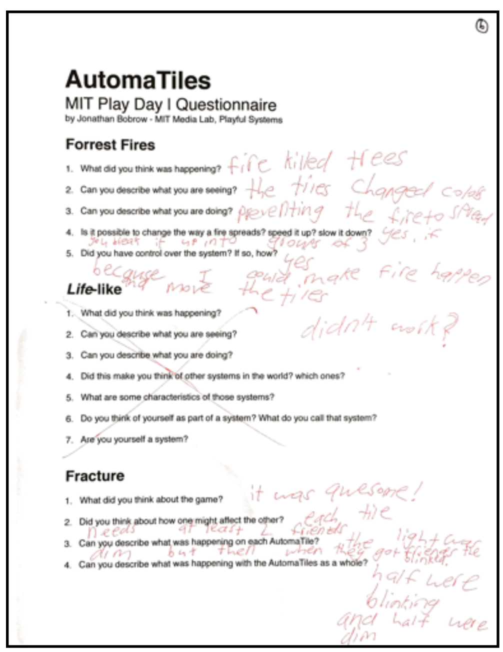
1. What did you think was happening? It caused the green tiles to turn into orange
2. Can you describe what you are seeing? It looks like a fire spreading
3. Can you describe what you are doing? You're trying to cause the fire to spread less quickly
4. Is it possible to change the way a fire spreads? speed it up? slow it down? Yes if you touch one lightening strikes.
5. Did you have control over the system? If so, how? Yes I did. I could cause the fire to grow or not grow.

***Life*-like**

1. What did you think was happening? I think they were trying to find which situation is best for each type.
2. Can you describe what you are seeing? The colors changing between red + blue.
3. Can you describe what you are doing? I was moving them around alternating between red + blue.
4. Did this make you think of other systems in the world? which ones? Yes - like people trying to make friends
5. What are some characteristics of those systems?
6. Do you think of yourself as part of a system? What do you call that system? Yes - the life system
7. Are you yourself a system? Yes

Fracture

1. What did you think about the game?
2. Did you think about how one might affect the other?
3. Can you describe what was happening on each AutomaTile?
4. Can you describe what was happening with the AutomaTiles as a whole?



AutomaTiles

MIT Play Day I Questionnaire

by Jonathan Bobrow - MIT Media Lab, Playful Systems

Forrest Fires

1. What did you think was happening? fire killed trees
2. Can you describe what you are seeing? the tiles changed colors
3. Can you describe what you are doing? preventing the fire to spread
4. Is it possible to change the way a fire spreads? speed it up? slow it down? yes, if you break it up into groups of 3
5. Did you have control over the system? If so, how? yes because I could make fire happen and move the tiles

***Life*-like**

didn't work?

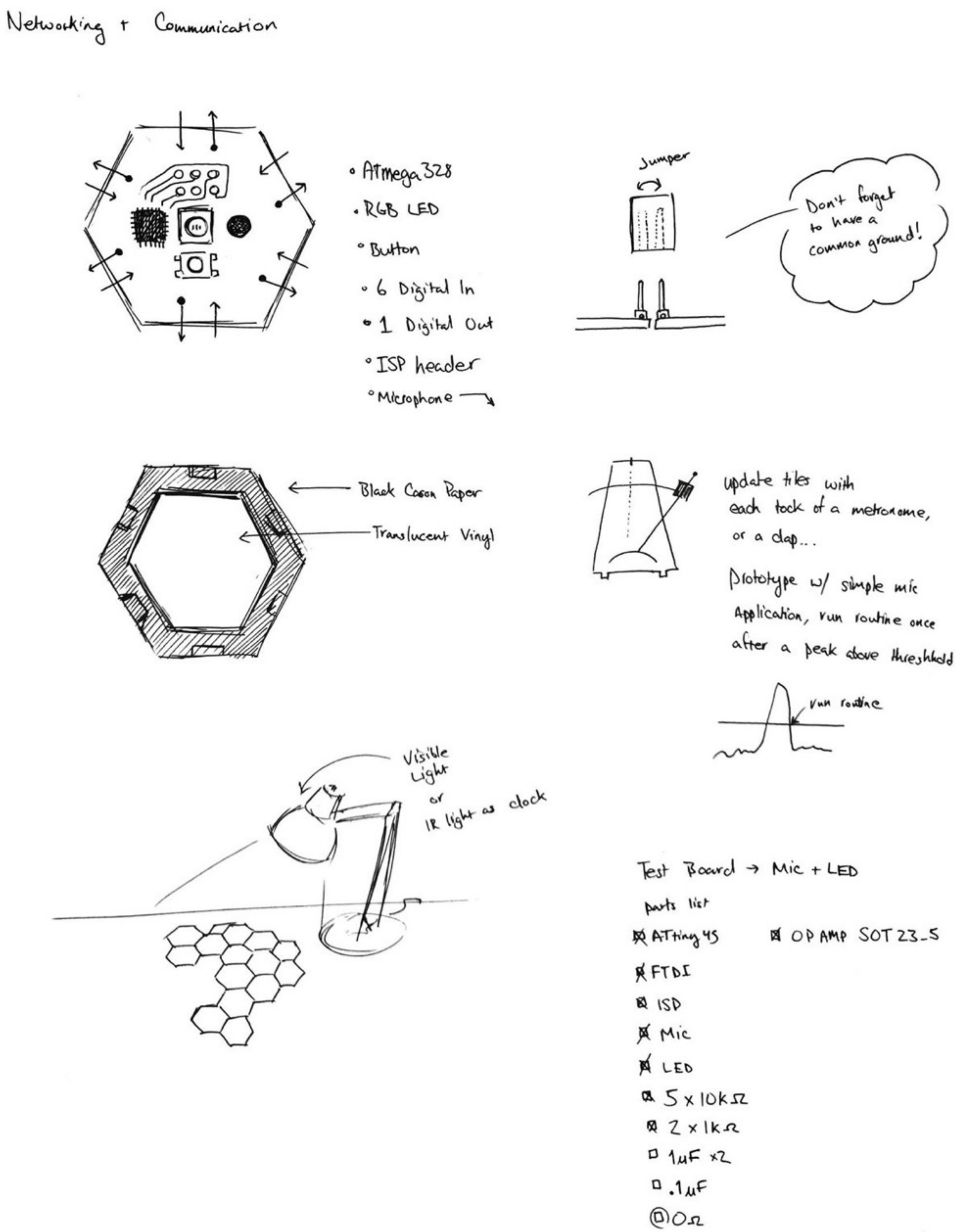
1. What did you think was happening?
2. Can you describe what you are seeing?
3. Can you describe what you are doing?
4. Did this make you think of other systems in the world? which ones?
5. What are some characteristics of those systems?
6. Do you think of yourself as part of a system? What do you call that system?
7. Are you yourself a system?

Fracture

1. What did you think about the game? it was awesome!
2. Did you think about how one might affect the other? each tile needs at least 2 friends
3. Can you describe what was happening on each AutomaTile? the light was dim but then when they got friends the blinked.
4. Can you describe what was happening with the AutomaTiles as a whole? half were blinking and half were dim

[25] please excuse the typo on my survey for forest fires, not corrected here since this is how the forms were received.

Appendix E


Networking + Communication
ATmega328
RGB LED
Button
6 Digital In
1 Digital Out
ISP header
Microphone
Jumper
Don't forget to have a common ground!
Black Canson Paper
Translucent Vinyl
update tiles with each tock of a metronome, or a clap...
Prototype w/ simple mic Application, run routine once after a peak above threshold
run routine
Visible Light or IR light as clock
Test Board → Mic + LED
parts list
ATtiny45
OP AMP SOT23_5
FTDI
ISP
Mic
LED
5 x 10kΩ
2 x 1kΩ
1uF x2
.1uF
0Ω