L-Systems: An Introduction to Generative Art

A Thesis
Presented to
The Division of Mathematics and Natural Sciences
Reed College

In Partial Fulfillment
of the Requirements for the Degree
Bachelor of Arts

R Jacob Hoopes

May 2023

Approved for the Division
(Computer Science)

Dylan McNamee

Acknowledgements

Thanks to my roommates, Henry Wilson, Brent Ellis, and especially Gabe Fish for the late night conversations that led to some of the more interesting choices made and questions raised throughout this document.

Table of Contents

Abstract

The world is a mess of systems. I hope to establish the context for one system that interests me and build it up for you so that you may be as interested in it as I have been. There will be an overview of the structure and function of L-Systems, which I will break down so that everyone can get a sense of what I've been doing. I'll explain the program that I wrote in detail and demonstrate some of its capabilities, as well as the steps I took to get where I ended up. I will also discuss some methods that others have used to do similar things, and go on to explain possible extensions to this work.

Dedication

This thesis is dedicated to the trees.

worlds and worlds and worlds and worlds

Chapter 1

Introduction:

[Illustration of a simple L-System, perhaps just a fractal tree]

The world of Computer Science can seem like a wilderness, teeming with frightening phrases like Zero-knowledge proofs, NP-Completeness, Finite State Machines, and the Halting Problem, but I hope to open up this world as one worthy of exploration. Specifically exploration at the hands of those from distant disciplines, like Anthropology, Economics, and Psychology, as well as the locals, pure Math, Bio, Chem, and Physics. The tool I will show you and give to you will demonstrate the accessibility of one of the most powerful, subtle, and yet strangely familiar disciplines: Computer Science. The ideas that I employ in the program have been given the name L-Systems, and their capabilities and beauty can astound.

Before we jump in, there are some things that must be said. This is a Computer Science Thesis, and so (essentially by necessity,) there will be code. I would like to keep this Thesis as accessible as possible to those who don't code, and so I have tried to design it so that anyone with the ability to feel wonder and joy can find something of value between these covers. The code will be accessible on my website, *rjacobh.github.io/website*. Some parts of the code of the central project will also be discussed at a later point in the thesis. However! I have included pictures on every page I could, and I'd like to imagine that they tell enough of a story by themselves that the words are ultimately unnecessary.

The fundamental structures that I will be discussing throughout this thesis are *Lindenmayer Systems*, usually called *L-Systems*. They are a type of "rewriting" system that have a distinct visual flavor. Introduced by Aristid Lindenmayer in 1968, they were first imagined as a framework for understanding the development of simple multicellular organisms. They were soon applied to larger systems, such as the structures of plants. A publication of Lindenmayer and others, *The Algorithmic Beauty of Plants*, was written in 1990 and has been an invaluable resource for me in the formation of this thesis. It introduces L-Systems as a tool to understand and revel in the beautiful complexity of plants, as well as to understand complex developments with limited arguments. Beyond this book, there have been a number of other vital resources in this process which I will discuss as they become relevant.

L-Systems are fundamentally a collaborative topic, they cannot be understood from the perspective of just one discipline. An idea of their structure and abilities can only be sufficient if a variety of fields are consulted. I've tried to reflect this awareness in the form of my thesis as well as its content. I've tried to write something that would be as interesting for an artist to read as a computer scientist. If you're here for the pictures, please enjoy! But if you're here for the discussion of the systems that create those pictures, I urge you to stay and engage with some of the questions that I've asked about life, computers, and the place this thesis has in that discussion. But regardless of what brought you here, I hope you have a fun time looking through this thesis, and it warms my heart to know that there are people interested in this work which has captured by imagination so completely.

[Illustration of an L-System, something that parallels the first one, and yet does something dramatically different (maybe the same image, just upside down? Inverted somehow?) Maybe just another cool L-System.]

1.1 Motivations

There are several motivations for this thesis, each which I hope will appeal to people from different disciplines with radically different interests. I hope that this thesis will be able to pull on threads in enough of a variety of disciplines that everyone will find something compelling in this short body of work.

The primary motivation, and the reason that I have become so personally invested in this topic, is the idea of *pattern*. Patterns that I demonstrate with L-Systems and other generative arts appear at different scales throughout life, the world, and our universe. I was in near constant conversation with my friends and acquaintances in other disciplines while I was writing this, and I was amazed at how often questions of self-reference, universality, and beauty from simple rules arise in areas beyond just computer science. I hope to draw in Economists and Anthropologists in addition to Physicists and those from other areas with discussion of patterns that appear in the structures of society and social organization, and which can be illustrated to some extent with the methods that I outline here.

The second main motivation, one that I try to emphasize at every opportunity, is the pure artistry behind so much of the patterns and methods that I explore. This is the motivation that I imagine might be the most interesting to artists, be they visual artists or otherwise. This is the reason I became aware of L-Systems and their relatives in the first place, so it is as essential a part of this story as the first motivation. It should be noted that it feels like the division between these fields is not nearly as dramatic as one might imagine. There are a huge variety of resources and tie-ins that connect the art aspects of this work to the computer science and math aspects, some of which I will explore later in the thesis.

The third motivation is tied in with a philosophical understanding of these structures. I am not a trained philosopher, and I would not be surprised to learn if many of the questions that I ask as a part of my exploration are in well-tread ground. It's possible the questions I ask in this thesis will contribute to the conversations where I imagine they might find a home, but I don't expect them too. This motivation comes from that observation that I mentioned earlier that there are some ideas that seem to exist in disconnected disciplines. Some of the questions I ask address larger issues of the place of machines in the creation of art and the creative process. It feels like these are essential questions to be asking when the wider culture is experiencing what may be the first wave of publicly available and popular AI systems. While this is absolutely not a formal foray into the philosophy of these questions, I do hope that my readers will take my suggestions seriously. I present a thesis which also confidently asserts the importance of these simple systems that a person can hope to understand, in a time when so many of the systems that control our lives are invisible to us. These sort of questions feel to me to be inseparable from the rest of the work.

I hope that there will be future interest in L-Systems, either as a result of the work that I've done here or as part of some other popular exploration of their abilities. This came to be a motivating project for me for more reasons beyond the three outlined above, and I imagine that someone who sets new eyes on this work could see it in a way I never could have foreseen. I go into more detail near the end of the paper about what I believe I have contributed to the conversation around L-Systems and their relatives, as well as what I believe the program I wrote could do if more time is given to its evolution.

1.2 The Path we will Follow

The first thing I will do in this thesis is formally introduce L-Systems. They will be explained and illustrated so that everyone who continues to read through my thesis will have at least some foundation for understanding. I will give different perspectives on the construction of these systems so as to help the reader see that they are incapable of being understood from any single one of these perspectives. The only way to get the whole picture is to know and appreciate that a single explanation will always be insufficient.

From there, I'll begin to demonstrate the abilities of L-Systems. They have a number of interesting capabilities, but the angle from which I've chosen to approach teaching about them is fundamentally visual. They are particularly striking when displayed in the way that I have chosen to display them, but there are different ways to go about doing this. Alternatives are discussed later in the thesis. While they can look very compelling, they've historically been locked in the jurisdiction of computer scientists. I wanted to make it so that people of any level of knowledge about them could engage with them in a way that could provoke wonder as well as teach, so I wrote a program to do just that. I go into depth about the structure of the program and some steps I took in its development, but I direct interested readers to just try the program themselves. How to access the program is detailed at the beginning of Chapter 2.

As I said, it's important to me that I keep this work accessible. To that end I direct readers to compelling resources that don't need any overhead, including my program. There are so many wonderful artists, mathematicians, and computer scientists creating massively interesting works that heavily relate to the topics I discuss here whose work I mention. I go on to talk about Generative Art and the broad crossover between art and computer science that has fueled this thesis and the work of many other contemporary artists.

However, beyond keeping this work accessible and hyping up Generative Art, I find it important to start to ask larger questions about what exactly it is that I've been doing. This section stands in my mind as the division between *what I have chosen to research and develop* and *what the consequences of my choices about research and development have been.* The thesis after this point will shift tone a bit to something more speculative. After I discuss in greater depth the other work that has inspired me to explore this niche, I ask some questions about the nature of my work. I also bring up some of the questions that I've asked or come across during my research. These questions poke at the foundations of my research and encourage readers to think critically about the content I'm sharing, in this thesis and in regards to engaging with art, machines, and the inseparability of those ideas. I find it especially important to bring the *why* of my thesis out into the open. I touch on contemporary issues like the rise of AI and the divide between robots and humans. As you might suppose, this is the part of the thesis where I suggest some of my takeaways from this research that touch on areas seemingly comfortably outside the realm of computer science. It is important to me to include this section, though I admit and acknowledge this goes beyond the scope of L-Systems and most of the rest of the work in this thesis.

After wandering a bit, the thesis returns to discussion of L-Systems. I explore the origin of L-Systems and their early connection to simulations of living structures, especially those of plants. *The Algorithmic Beauty of Plants*, or *ABOP*, written by Aristid Lindenmayer, their progenitor, and Przemyslaw Prusinkiewicz, has been an invaluable resource for me in the development of this thesis. In this section, I touch on some of the ways that this book guided parts of my research and helped me make judgement calls about what I'd be focusing on. I talk about why I chose to include what I did and why I refrained from integrating some of the discussed ideas into this thesis before this section.

Here I discuss possible extensions of the program. Some of these extensions are little more than quality of life changes while others might require a total rewrite in order to function. They all hope to pursue one or both of two goals. The first goal is increasing usability and user-friendliness. The second goal is increasing functionality. In order to more competently think about the second of these goals, I reference the choices I've made regarding the content of *ABOP*. I address some of the considerations that might have to be made in order to introduce some of the structures in *ABOP* to my own program.

This last section focusses on the limitations of L-Systems. First, I try to generalize L-Systems and clearly articulate their capabilities. I discuss dimensions and the way that L-Systems engage with different dimensions. I suggest possible alternative dimensions that could be explored and ways in which L-Systems displayed with those dimensions could look. The problem of depicting some of these theoretical dimensions will also be explored here, as well as why expanding understanding of L-Systems could be worthwhile.

I then conclude the thesis with a brief summary of the understanding I've gained through this process, the concrete objects that I have pulled into being, and the exploration of the larger ideas that lie under this whole process. I restate a few suggestions about how L-Systems might be investigated in the future and what there might still be to be learned about these structures. I end by encouraging my readers to mess around with my program, explore L-Systems and the larger world of Generative Art, and to be excited by work like this.

Chapter 2

What is an L-System?

This is an L-System:

[Picture of impressive looking L-system]

Well, it would be more right to say that this is the *output* of an L-System. L-Systems are a *technique* with which one can describe a complex, changing system with just a few letters and symbols. Some Biologists may take offense, but it may be compared to DNA in how it compresses a potentially infinite complexity into a handful of characters. This chapter will cover a few main ideas. The first, and most central, is a discussion of the mechanics of how they work and how to build them. After that, I'll go on to describe some of their main features, some of their other traits and patterns, and their applications. As part of that last point, I'll illustrate particularly effective ways these systems can be used, how I've used them, (including some early experimentations before the central project,) and end with a lead-in to the central coding portion of this thesis, the L-System Builder.

There are a few effective ways I've found to introduce L-Systems to people. I'll go through each of these different strategies to give you some solid footing before we really dig into the material.

2.1 Understanding through Definitions

L-Systems have three main parts. The first part is the "start" variable. It is usually just a character, we'll call it "A" here. This will be the initial value of the structure, something akin to a "seed". The second part is a set of rules that describe how the L-System changes. These are usually written like "A $\rightarrow$ AB" or "B $\rightarrow$ AA", where each of these is a different rule. There are different names for what "A" and "B" are, but in this thesis I will call them "Productions". In a similar fashion, each of these rules is called a "Production Rule". (Any character without a corresponding production rule is therefore not a production. This will be relevant later.) The third part of an L-System is a list of other information that is used when it is being constructed. For most of the L-Systems discussed in this thesis, there are two pieces of information in this list, a value θ which is read as the "angle" and a value "Iterations" which is the number of times the rules are applied to the system. Alternative members of this list are discussed in Chapter 5.

Before I describe the process of making an L-System, I'd like to explain some vocabulary that I'll be using throughout this thesis. There are a few ideas that will hopefully make understanding these systems much easier to understand. The first term that I'd like to teach you is *string*. A *string* in computer science is a more expansive way of saying a *word*. A string does not need to be in any dictionary, however, and it can include essentially any character that can be written on a computer. This includes numbers, symbols like "&", "%", and "+", as well as some other special characters. When I say "string", I mean a word that consists of some combination of these characters, for example: "DFS&2*—:09" or "!!n./?;". Strings do not need to be able to be spoken aloud to be valid. A string can also be "empty", meaning that it contains no characters, though it is still a string. They may also be only one character long.

To draw an image from an L-System, we follow a set of carefully stated instructions. First, we save the start variable as a special string, we'll call it the L-string. If the start variable was "A", then the L-string is also "A". Then, we repeat a specific step however many times as determined by the Iterations value. If the Iterations value is 5, then we will repeat the following step 5 times:

> - For every character in the L-string, if that character has an associated production rule, then that character is replaced in the L-string by the production rule. (For example, if the L-string is "A", and there is a production rule "A $\rightarrow$ BA", then the new L-string after this step is "BA".) If that character does not have an associated production rule, then it is left alone.

With this simple step as a guideline, we can construct even the most complicated L-Systems. It might seem strange that this one rule is capable of such complexity, but that surprise is the mood that I'm encouraging you to see as a fundamental feature of these structures. There may be more opportunities to be surprised!

2.2 Understanding through Images

By using this sort of image to describe these structures, I define the dimensions in which they operate and therefore lose access to some of their possible complexity. I suggest alternative visualization strategies in Chapter 5. [A series of images that show the progression of an L-System through multiple generations. Each of the images is labelled with a short description that explains the process shown in that image.]

This set of images is generated with the parameters, [start: A; rules: A $\rightarrow$ AB,

B $\rightarrow$ AA; θ : 45°]. Different values for iterations are displayed in each image. “A” is drawn as a line going upward and “B” is drawn as a line going to the right. Each new production starts from where the last one left off.
[Image A0]: Iterations = 0, current string: A
[Image A1]: Iterations = 1, current string: AB
[Image A2]: Iterations = 2, current string: ABAA
[Image A3]: Iterations = 3, current string: ABAAABAB

Here is a different set of images with the same start variable and extra information, but alternative production rules. These rules are: A $\rightarrow$ BAB, B $\rightarrow$ A.
[Image B0]: Iterations = 0, current string: A
[Image B1]: Iterations = 1, current string: BAB
[Image B2]: Iterations = 2, current string: ABABA
[Image B3]: Iterations = 3, current string: BABABABABAB

These

2.2.1 Turtle

The programming language Python has a built-in library called “Turtle” that enables users to easily write code that draws simple diagrams. There isn’t much overhead required to begin experimenting with Turtle, a fact which is reflected in how often it is used in introductory computer science courses to get new computer scientists used to the idea of creating images with code. It is also particularly effective at teaching people about the results of changing their code. Part of its quality in this regard is how it animates its drawing. Turtle was one of the early methods I used to experiment with L-Systems. The images in the rest of this section show different L-Systems that I’ve drawn with Turtle.

The essential feature that Turtle uses to great effect is “state”. “State” here means “condition” or “status”. The Turtle keeps track of a few attributes as it goes through drawing the image described by the code. The attributes that are essential and which I will focus on here in my larger discussion of L-Systems are the Turtle’s position and its rotation. With an x-position, a y-position, and an angle, the Turtle is uniquely placed in its window. By manipulating these values with the code and choosing whether to draw or not draw between positions, the Turtle is able to draw a single path through every line that it draws.

Chapter 3

Explaining the Program

In order to share my understanding about and my excitement for L-Systems, I wrote a program that allows users to directly change the parameters of the system without having to directly engage with any code. My central goal was to create a program that would reduce the barrier to entry. My first explorations into making this space more accessible were done through turtle implementations in a basic python program and later in a blender program. Both of these strategies still heavily relied on the user to type input that might as well be code. It became clear that if I was to create something that actually met my goals that I would have to change my approach. In conversation with my advisor and through resources online for projects like mine, I came across *Processing*.

3.1 What is *Processing*?

Processing is a programming language built off of Java that has been made especially to work with image generation. It was released by Casey Reas and Ben Fry in 2001. I chose to work with Processing because it was easy to use and

3.2 Program Development

The development of the program that I would eventually come to call the L-System Builder started with a series of drawings describing the different functionalities I was aiming to include.

3.3 Access and Use

The program that I've created for this thesis demonstrates the structure of an L-System and allows the user to interact with the parameters to dynamically adjust the displayed content. This “L-System Illustrator”

L-Systems are a way to see every stage of the life-cycle of a tree simultaneously. There are limitations to this metaphor, but the freedom of understanding that is gained with the metaphor more than makes up for the rough patches. It’s also accurate especially by the use of the “tree” metaphor.

Chapter 4

Contribution

A primary goal of this work is to make L-Systems, and by extension, generative art and even Computer Science as a whole, less frightening. I want folks who have never used computers become able to create things on their own initiative to be amazed and inspired by the possibilities. I even want my peers, who have been immersed in the depths of CS for years, to be able to approach the discipline with a fresh awe and understanding. I hope that this will spark something like that in my readers, though even just using the resources I've collected and attach to pursue your own interests is a dreamy enough prospect.

I am fully aware that L-Systems have been around for 55 years, since they were proposed back in 1968, and for a time I was worried that with 55 years to develop these ideas I wouldn't be able to tread on any new ground. This is true in many respects, and I re-emphasize that a main goal of this work is to raise awareness of L-Systems and to bring together sources that discuss these ideas. However, this work goes a little beyond that. There are few implementations that make L-Systems easy to play with. Even Nikole Leopold, a grad student at the Technical University in Austria who made a brilliant program to bring

No one has made L-Systems this easy to play with as far as I'm aware. Despite the central part of their appeal being their simplicity I haven't seen any implementations that allow people to play around with them freely. - so few characters, and yet there's been no way for people to engage with them easily. How to gain an intuition.

A central contribution of my own has been the development of the interactive L-System Builder

4.1 Related Work

I've written three main programs for the generation and display of L-Systems, a native Python program, an implementation of that program into Blender, and the final L-System Illustrator that I've polished and for which I've added some instructions.

These projects were interspersed with a research process that involved investigating other implementations of L-Systems and related display systems. I'll discuss a few of those briefly here before expanding my view to include other visual structures that I feel are worth exploration.
A useful resource for me has been the YouTube channel *The Coding Train*, and I'm grateful for the wide variety and expansive content of their videos. They were my introduction to some of the ideas that I talk about more in the following Generative Art section as well as much of my foundational learning about Processing.
A graduate student at the Technical University in Vienna by the name of Nikole Leopold wrote a Bachelor's thesis about an implementation of L-System graphics in Blender in 2017. I would have liked to come across this paper earlier in my process, but it still has been able to inform some of the later discussions that I've had as part of this thesis. Leopold gives a comprehensive overview of the advantages of using a 3d modeling software like Blender to display L-Systems. One of the pieces of that research that I found especially interesting was the inclusion of methods by which the L-System can interact with its environment. The structure grows, finds an obstacle, and engages with it in such a way as to appear to go around it. (The program actually just cuts off the branches that intersect the object.)
There are other extremely interesting and visually very appealing papers by other scholars. A central

4.2 Generative Art

One major subset of those creations fall under the label *Generative Art* and can be investigated by just searching for that online. I've found online communities that create beautiful streams of this work while only searching a little bit. I've also been fortunate to be near many people that are also interested in many of these same ideas and have supported me in this work more actively. I credit Trever Koch for helping me bring many of my early ideas to life. The point here is that there are resources available if you are at all interested in learning more about this. I have collected specific resources and put them in Appendix B for your perusal and easy access.
Include references to some of the numerous types of generative art that I've come across during my research.

4.2.1 My Explorations

Show some of the different projects that I worked on in the build-up to this thesis. Explain how they led into L-Systems or any other part of my work that eventually coalesced into this thesis.

4.3 Philosophical Quandaries

Here is where I feel it's appropriate to ask some of the big questions. Start with an intro explaining and defending the existence of this section,

4.3.1 Is Math Art?

Is this art? Where do we draw the line between art and math?

4.3.2 The Times We Live In

What does it mean for a machine to make art?

Chapter 5

Past, Present, Future

L-Systems were introduced in 1968 in a paper by Aristid Lindenmayer, where he suggested how one could construct a mathematical model of the structure of simple cells. Even in their introductory paper, L-Systems were deeply tied to the modeling of life.

5.1 The work of Przemyslaw Prusinkiewicz and Aristid Lindenmayer

Use this section to talk about the wide variety of other forms of L-Systems that I have not investigated in this Thesis. I will make mention of parametric L-Systems, stochastic L-Systems, context-dependent L-Systems, and L-Systems that are used for a wide variety of other uses. From tree-modeling at Pixar to demonstrating cell-growth in biology labs. Say essentially that *The Algorithmic Beauty of Plants* has an abundance of thoughtfully demonstrated resources and examples, as well as many compelling pictures.

There are some compelling parts of the book that go well beyond what I discuss in this thesis, specifically regarding extensions to L-Systems that dramatically increase their capabilities and functionality.

The Algorithmic Beauty of Plants

5.1.1 Early Uses

5.2 Extending the Program

Talk briefly about different ways in which the program could be extended to address what I perceive to be flaws or oversights. Mention things that I wished I had time to change, but go beyond this to discuss features that might deserve their own program.

Include a lead-in to the next section

5.3 Generalizing L-Systems

So far, we've only been looking at L-Systems that fall into a very narrow category. Prusinkiewicz and Lindenmayer go into detail about a number of variations on L-Systems that increase their complexity but also their capabilities. I've avoided these extensions for two reasons, one reason being time and the other being a desire to keep the complexity of this thesis low so as to invite people in who might otherwise feel intimidated. I'll focus on three main variations that were discussed in length in *ABOP*, and I invite the reader to explore them if you feel so led.

The first of these systems is context-sensitive L-Systems. In the L-Systems that I've shown up until this point, each production variable does not depend on the surrounding values when determining what its production will be. If we choose to explore what L-Systems can look like when they are context-sensitive, we can see straightaway that the possibilities will be more expansive than with the context-free L-Systems we've been seeing up to this point. Any basic context-free L-System can be made by a context-sensitive L-System where none of the productions actually depend on context. This type of L-System can build structures that look more like something you might expect from a cellular automata, like John Conway's Game of Life.

Stochastic L-Systems are a type of L-System that use randomness to construct different structures each time a particular L-System is drawn. These are interesting in a couple ways, but I find it most interesting that they seem to contradict the fundamental argument for why L-Systems are compelling. That reason being the rise of apparent complexity out of a set of simple rules, and specifically complexity that is deterministic. There is something interesting about how these might demonstrate the effects of mutation in an organism, though I'm sure that

Parametric L-Systems are one of the most personally appealing and powerful variations on this

5.3.1 What does it *Mean* to Change Dimensions

When the state at a particular point in the construction of an L-System changes, it alters the way that the L-System is displayed. This can be especially obvious and dramatic if the part of the state that is changed is the angle. By changing the angle by only a small amount, let's say a degree, we affect the position and rotation of every following node. Changing the position part of the state will only change the position of later nodes. This seems to suggest that somehow, changing the angle is a *more powerful* alteration than changing the position. Looking more closely at what actually separates changing position from changing angle, we can see that the angle never actually draws anything. The value that draws things is always F. We can take our simplified L-System and see that there is a depth of nuance to the values we implicitly chose to use at the beginning of this process. The following table shows the differences between these two operations, the rotate operation and the forward operation.

Table 5.1: Rotate vs Forward Atomization

Attribute	Following Node has the Same Position	Following Node has the Same Orientation	Draws
Rotate	Y	N	N
Forward	N	Y	Y

Every time the Forward operation is read, two things happen, while when the Rotate operation is read, only one thing happens. It seems like it might be possible to create a more specific operation for forward that breaks it up into its two parts. The way that people have found to get around this problem is by introducing a second type of Forward operation that changes position but doesn't draw. Usually this is notated by a lowercase f instead of the usual capital F for the regular Forward operation. We could approach the forward operation in a different way - perhaps it would be more right to call it a "draw" command, and the way it draws is governed by either the state of the L-System at the moment when the operation is read or the parameters of the L-System at the outset, or both. If we begin to think about the forward command as a draw command instead, a new world of possibilities opens to us.

Imperative vs declarative (turtles are the ultimate imperative thing) L-Systems are fundamentally declarative - just the replacement system Don't think about going through the string piece by piece, instead just showing the before and the after

Declarative language "prologue" - rules are defined & a hidden engine solves with attention to those restraints

5.3.2 Alternative Dimensions

What if we choose to change our chosen dimensions to display the same L-System in a different way? There are a few ways to do this *I think).* First, we must figure out possible alternative dimensions. Beyond the basic three spatial dimensions, a classic dimension is time. This suggests that there might be a way to display L-Systems with variations in time, perhaps in the form of an animation. Some other classic dimensions that people have used with L-Systems are width and length (as in determining how wide or long each line is drawn). We could change the color of the line, so that it changes hue, saturation, or opacity according to some color space. We could alter the curvature of the line, perhaps in such a way that we have a smooth, shamelessly organic-looking system. The quality of the lines could be a dimension: we could make them more jagged in one direction and more smooth in the other.

5.3.3 The Problem of Depiction

Some dimensions lend themselves more easily to being displayed. For example, if we try to display an L-System which has a draw function that overwrites previously drawn sections, then we lose information. This might not be bad if that's gives an effect that we're attempting to capture, but we are still losing information. As another example of this same problem, and hopefully one that's more concrete, when the angle is set to 0 or 180 in the Illustrator, you'll notice how the resulting L-System is just a vertical line. Much information is lost. We can imagine adding color to the Illustrator in such a way that each branch segment has a color, but this still will lose information, though not as much. If we see the *problem of depiction* as the problem of losing information, then we can try to solve the problem of depiction by somehow retaining all the information. A way that could work is extending the program into three dimensions, where each line increases in depth over the course of its run. We would have to be able to move in such a way as to be able to see this change, but it seems possible. We could then see that later branches are deeper into the screen (or higher out from the screen). Either way, we've seemingly solved some of this problem. We also might be able to do the same thing by reducing the width of later branches and changing their color. We'd then be able to pick out the later branches from the earlier ones, retaining some information and seemingly resolving this problem.

Both of the examples I gave address the issue of losing information by simply adding more dimensions. The added dimensions can be spatial or they can be in the form of attributes, but either way they add something that makes branches unique.

Chapter 6

Conclusion

This thesis has been a joy to put together. I've reveled in each opportunity to engage with the material and explore more about this idea whose depth I feel I have only begun to understand. Engaging with my peers and mentors about L-Systems and Generative Art has opened my mind to new ways of thinking about these ideas, not to mention the joy I feel when they respond with their interests in kind. I've attempted to make this thesis multidisciplinary, in the spirit of L-Systems as a child of art and computer science but also in the spirit of Reed College as a Liberal Arts institution. I have found that no idea is complete without a range of perspectives.

L-Systems are

Appendix A

Extra Images

Appendix B

Generative Art Resources