Computational Aesthetics in Graphics, Visualization, and Imaging (2011)
D. Cunningham and T. Isenberg (Editors)

Double Meandering Algorithm: From Drawing Game to Automated Animation

Shelley Gao , Lucy Pullen, and Amy A. Gooch

University of Victoria

Abstract

We introduce artist Lucy Pullen's Double Meandering Algorithm, first in its original form as a pen-and-paper drawing algorithm and then as a procedurally generated animation. We utilize a chain of cubic Bézier curves to represent the characteristic spiraling line, assigning each control point according to a pseudo-randomized algorithm. The resulting curves are then animated segment by segment, reflecting the artist's process of creating the pen-and-paper drawing. By digitizing the Double Meandering Line drawing, we can also reveal the process of creation through animation, granting us the ability to exhibit a fundamental part of the drawing that is lost in the traditional pen-and-paper presentation.

Categories and Subject Descriptors (according to ACM CCS): J.5 [Computer Graphics]: Computer Applications—Fine Arts

1. Introduction

We present a digital representation of a pen-and-paper 'drawing game' called the Double Meandering Line. The Double Meandering Line is the novel creation of Lucy Pullen, professor and conceptual artist based in New York and Victoria, British Columbia.

The Double Meandering Line is a form of generative art based on a set of predetermined rules; a finished drawing is simply an outcome of following simple two-dimensional rules for laying down each iteration of curves. The resulting shapes, however, appear to be three-dimensional after completion. A key feature of the drawing is the process, not just the context of the finished work. To experience the contrast, the viewer must witness the drawing in the making – a luxury that is not afforded by the traditional gallery model. Thus, we endeavor to create a procedurally generated animation that bypasses the inherent limitations of both static 'finished' art and process art.

Our procedurally generated animation, the Double Meandering Algorithm, attempts to capture the three-dimensional effect achieved by the artist by combining the same sort of segment drawing rules with chains of cubic Bézier curves and pseudo-randomized variables to generate a digital drawing with a similar aesthetic.

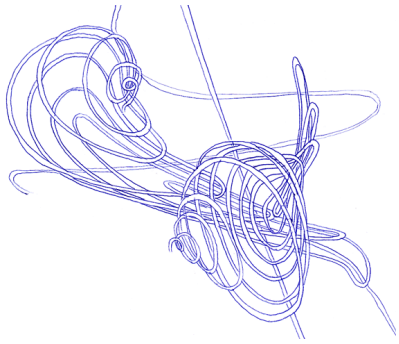

Figure 1: *An analog Double Meandering Line drawing by Lucy Pullen serves as the basis for our digital algorithm.*

In this paper we report discoveries made as computer scientists try to blend their skills with the talents of an artist to create a digital version of an artist's "drawing game", and to reveal the process through animation.

2. Related Work

Our shared interest as researchers in art and computer science is in new imagery. To make new images, we have taken

CAe 2011, Vancouver, British Columbia, Canada, August 5 – 7, 2011.

DOI: 10.2312/COMPAESTH/COMPAESTH11/055-062

a process of our own creation and are attempting to recreate it in another domain. This job exceeds the scope of a single creative individual and discipline. We are working with inside information on a particular artistic method in the domain of computer science.

Our primary goal is to emulate the results of the Double Meandering Line moreso than the process of it – we break pre-existing rules if we find a better solution. However, the artist is actively involved in the planning and development of the procedurally generated animation described in this paper, making the system a collaborative work rather than a derivative on the part of a research group.

2.1. Background in Visual Art

Guidelines for technical drawings of industrial objects were first described by the French Academy in the late 1800's. In her essay "The Language of Industry", Molly Nesbitt discusses how Marcel Duchamp used these rules to create new modes of abstraction [Nes86]. Common objects such as the shovel and the urinal appeared as art object Readymades as early as 1917, thus breaking the classifications established by the French Academy.

It is accepted practice in both Minimalism and Conceptualism for the artist to be removed from the physical construction of the finished piece. Minimalist Donald Judd had sculptures constructed by factory contractors, and Conceptual artist Lawrence Weiner produced pieces that consisted of nothing but short instructions, which could stand alone or be carried out by gallery staff onsite [Hop00]. In both cases, the only name attached to the creative aspect of work is that of the artist, and we adhere to this convention here.

Generative art is associated with practices where the artist cedes control of the outcome to a self-supporting system, such as a set of rules or a computer [Gal03]. Though the term itself is a 20th century invention, the principles have been in use for far longer – the definition of a self-supporting system could be as broad as to include the use of concepts as universal as symmetry. The movement encompasses not only visual art, but also music and literature. It is one of the principal movements to deal with the element of randomness, which can often be the main driving force of the work.

As computers are one of the primary platforms for self-supporting systems at our disposal today, a good portion of computer-assisted art falls under the generative art heading. Additionally, web application platform such as Flash and Processing have given generative artists a convenient medium in which to develop and package their work. For example, Jared Tarbell maintains a collection of his algorithmic art on the internet, complete with source code [Tar10].

2.2. Background in Computer Graphics

Much research in the fields of Computational Aesthetics and Non-Photorealistic Rendering originate in the emulation of the personal styles of visual artists. For example, work by Lee *et al.* introduces a fluid jet simulation that allows a user to produce pictures in the style of Jackson Pollock [LOG06]. Xu *et al.* developed a system that replicates traditional Chinese ink paintings as digital strokes, which can be manipulated to create animations based on the paintings [XXK*06].

Our work as a programming artifact has intents in common with generative art and computational aesthetics [Gre05], but our Double Meandering Algorithm is heavily based on the traditional drawing rules of one specific set of works by Lucy Pullen.

3. Double Meandering Line Drawing

A firm understanding of the Double Meandering Line is a prerequisite to the development of the digital algorithm. Though differences between the analog and digital processes are inevitable, we want to adhere as closely as possible to the original process. The Double Meandering Line is not generative art, but the algorithm is generative by necessity. We concentrate on the generative art principles of the Double Meandering Line in our analysis, as it is the most effective paradigm for our purposes.

Complexity in the Double Meandering Line is the accumulation of many simple parts. The drawing consists of two parallel lines following a curve, from the beginning point of each spiral to the end without touching. Together they create the impression of breadth in a unified object. Each pair of curves comes back to near where they started and once the lines intersect, they stop and then both lines continue on the other side; one leads and the other follows, as if forming a knot or length of rope. However, this is not a drawing of a knot as explored by Kaplan and Cohen [KC03]. This is a simple set of rules, creating a deliberate manufacture of coherent abstraction.

In this paper we present a study and implementation based on a simple, early version of the Double Meandering Line as the basis of our system, presented in Figure 1.

3.1. Terminology

First, we will describe some terminology that will be used throughout this paper. We define each loop of the drawing as a **curlicue** and the process of drawing each curlicue as an **iteration**. A **spiral** can be described as a series of **curlicues** with connected ends. We define the **spine** of the spiral as the imaginary line on which each curlicue is strung.

The curlicues of the spiral define the mutual path of two parallel lines. The line drawn on the inside edge of the spine is always the first one to be drawn, so we will refer to it as the **original line** and the outside line the **following line**. Through parameterization and randomization we control the **spine** or offset path of the curlicues onto the page, generating the 3D-esque shape found in the Double Meandering Line.

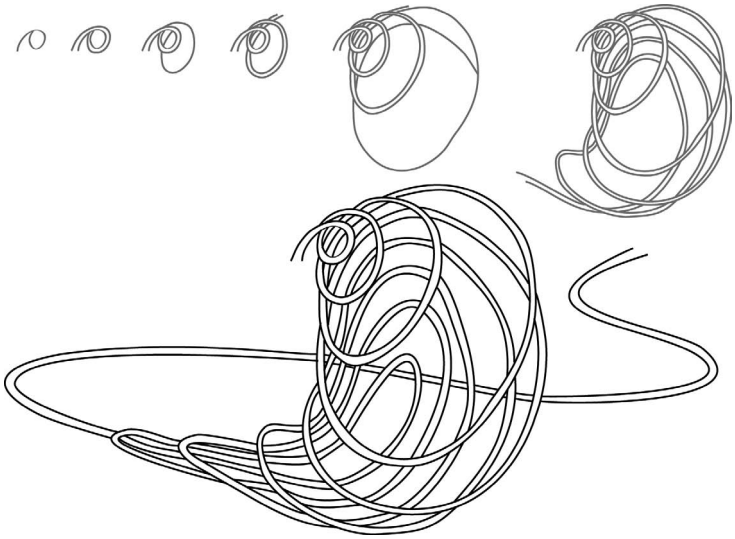

Figure 2: *Progression of a Double Meandering Line drawing spiral from start to finish, by Lucy Pullen.*

3.2. Step by Step Analysis

As a "drawing game", the Double Meandering Line requires no tools save for a drawing surface and a pen. Though more complex, post-spiral Double Meandering Line drawings can span large swaths of paper, the technique is scaleable to any size. Generally, two to four variously sized and shaped spirals are drawn on the same page, overlapping each other. The combination of multiple spirals furthers the sense of depth and adds complexity to the composition.

The process of creating a Double Meandering Line of the spiral type is a straightforward, step-by-step process. The Double Meandering Line has two stages: an **Opening Stage** and a **Closing Stage**. The Opening Stage starts with a self intersecting curlicue, with the following line running parallel to the first and ending upon contact with the original line. The next curlicue is a larger version of the first iteration, with the original and following lines starting close to, but not exactly, where they left off. They start from a point of contact with the following line rather than exactly where they left off, illustrated in Figure 2. This creates an illusion of overlap and depth as if the double lines are a single entity, similar to the 'rope' found in Celtic knots.

As the curlicues increase in size, irregularities occur in the shape of the curve – an accidental bias towards one end of the curve, for example. Instead of correcting for the previous irregularities, the curlicues exaggerate the characteristics of these irregularities as they grow in size. The most exaggerated of these irregularities forms a visible bias and is called the **attractor**. The attractor is the main feature of the Closing stage, where the curlicues cease to increase in size. Instead, the curlicues begin to flatten and taper towards the attractor. This continues until the curlicue becomes thin and elongated to the point where it cannot contain the next iteration. Then, the line stops iterating and trails off in the direction of the tail. The result is a conch-like shape that appears to be made of thick wire.

Double Meandering Lines depend on the interrelation between an iteration and its predecessors. The shape of each iteration can be predicted by the previous iteration, but cannot be fully explained; the irregularities of the predecessor are amplified with each iteration.

4. Procedurally Generated Animation

From the analysis of the Double Meandering Line, we gather a set of requirements for a program that will emulate the drawings with reasonable accuracy. The computer algorithm's success lies in the convincing emulation of a spiral shape, balancing randomization with authenticity.

We require a structure that easily accommodates a smoothly spiraling line. The structure should be aware enough of the shapes of the previous iterations to imitate the overall shape, while also having the flexibility to change drastically between iterations.

For our initial algorithm we considered two-dimensional spirals, such as the logarithmic spiral, since spirals are already similar to the shape of the Double Meandering Lines and can be described using a single polar coordinate equation. However, they are not sufficiently malleable – one cannot independently warp the shape of a particular iteration without affecting the shapes of the preceding or succeeding iterations. The shape of the spiral iterations can only be modified via a global matrix warp, which cannot warp a single iteration of the spiral without affecting the others.

Instead, we take advantage of the flexibility and controllability of drawing with splines. The use of splines is inefficient compared to the use of spiral primitives, as even a single Bézier curve is composed of multiple equations. A chain of Bézier curves also requires storage for each and every node position in the chain. Regardless, accuracy of shape takes precedent over speed in this case. A chain of Bézier curves, linked from end to end, creates a malleable curved line that can serve as our spine. Though the spiral arrangement must be created from scratch, the placement of each node is completely independent from all the others. Bézier curves also make it easy to animate a drawing line because the curves are rasterized as connected line segments.

In our algorithm, we first determine the control points of the curlicues and the parameters necessary for determining the spine of the spiral in the shape calculation stage. The original and following lines are then drawn to the screen in the animation stage.

The first version of the Double Meandering Algorithm was developed using the Processing API. It has since been ported to iOS devices in Objective C.

4.1. Structure

We generate each iteration of curlicues using four cubic Bézier curves, labeled C_1 though C_4 (Figure 3a). Each cubic Bézier curve is modified by four control points, which will be labelled cp_1 through cp_4; cp_1 and cp_4 represent the endpoints of the line, while cp_2 and cp_3 are used as tangent vectors that control the trajectory of each end of the curve.

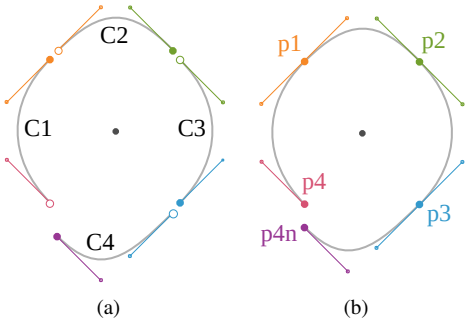

Figure 3: *The structure of a single curlicue, or "iteration". a) The four independent Bézier curves used to build the loop. The filled cp_4 share the same coordinates as the unfilled cp_1 of the previous curve. b) The five mutual points that occur in an open-ended curlicue, with the first and last points being mutual points with the preceding and succeeding curlicues.*

Thus for each iteration we need to assign coordinates to 16 control points, four per each curve. Each curve shares its cp_1 with the cp_4 of the previous curve and its cp_4 with the cp_1 of the next curve, thus the duplicate endpoints that can be treated as a single entity we call p_i, for $i \in [1,4]$ (Figure 3b). The new points, p_1 through p_4 (and p_{4n}, representing the p_4 of the next curlicue), are also attached to two tangent control points, i.e. the cp_3 of the preceding curve and to cp_2 of the next curve. The coordinates of these two control points are always set to be the reverse of one another, thus maintaining C^1 continuity between the previous curve and the next. We note that a higher order spline curve (such as a NURBS) may replace the four curves, however here we document our original approach. The spiral is defined by at least six iterations of curlicues, depending on how many iterations are in the Opening Stage.

4.2. Shape Calculation

In the first stage of our algorithm we set up the random variables that parameterize the control points and spine of the spirals. First, we randomly choose the main direction for the spine of the spiral based on one of three types, illustrated in Figure 4. Each spiral type is based on which curlicue point acts as the attractor: p_2, p_3 or p_4.

Next we initialize the variables of our spirals with randomized values within the ranges listed below.

- Seed coordinates of the spine, i.e. center position of the first curlicue, C
- Radius of the curlicue, r, a value between 0.5% and 4% of the screen width
- Additional length of spine, between 1% and 2% of the $r * i$, the current iteration
- Growth of radius during growth iterations in the Opening Stage, between 60% and 140% of r
- Number of growth iterations in the Opening Stage, between 1 and g, for $g \in [3,5]$
- Attractor tilt shift, between -797.5% and 12.8% of r, depending on spiral type

The seed coordinates for the center of the first curlicue are restricted to values that keep the center of the spiral on the screen. Additionally, for the first spiral, we further restrict its position and size such that the whole spiral fits on screen, taking into account the size and type of spiral. We relax this restraint for subsequent spirals, creating a parameterization such that a satisfactory percentage of the spiral will be drawn on the screen.

The drawing of a spiral consists of two phases, similar to the process described by the artist: growth iterations (Opening Stage) and warp iterations (Closing Stage). During the growth iterations (or the opening of the spiral), the radius of each respective curlicue increases according to the initialized radius growth multiplier. A spiral consists of a randomly determined number of growth iterations between 1 and g, for $g \in [3,5]$. In each growth iteration, we process the Bézier curve control points of the associated curlicue. The control points are initialized to lie in a circle of radius r around the current curlicue center, C. Then, we alter the position of the

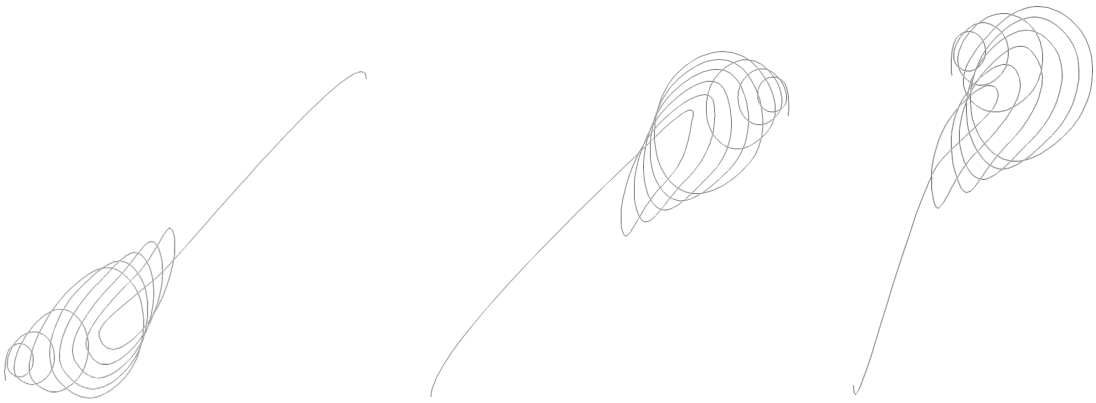

Figure 4: *Three basic types of spirals define the direction of the spine of the spiral, according to an attractor point.*

control points based on the attractor tilt shift, as shown in Figure 5. Finally, we increment the center of the curlicue to the next coordinate on the spine and, if this is a growth iteration, increase the radius.

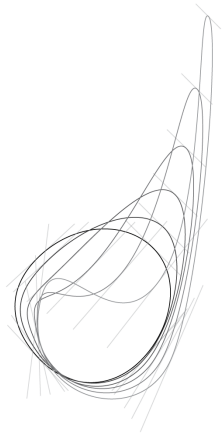

Figure 5: *A test case for warping towards an attractor. In this case, the attractor is* p_2.

During the warp iterations, or the Closing Stage, the radius ceases to grow, but the spine changes trajectory and the shape of the curlicue becomes more and more exaggerated towards the attractor. We found that algorithmically determining the number of warp iterations in the Closing Stage was difficult. Through trial and error, we determined that six warp iterations followed by the tail most closely replicated the behavior of the spirals found in Lucy Pullen's drawings. The basis spirals already have some bias towards one side of the attractor as opposed to the other. If pulled too far against the direction of the tangent tilt, the shape of the spiral will break down. The lopsided attractor tilt shift modifier values account for this problem.

When we reach the end of the warp iterations, we use one last iteration to make a tail. We achieve this by setting the control points before the attractor like a regular warp iteration. Then, we multiply the coordinates of the attractor by a scale factor of suitable magnitude (default setting is 300%), and set all the control points after it to those same multiplied coordinates. This gives us a 'tail' that follows the same trajectory as the attractor.

4.3. Drawing and Animation

The Double Meandering Line begins in the first dimension with a point, graduates to the second dimension with a line, and to the third dimension with a second line. The digital Double Meandering Algorithm takes this progression one step further towards a four dimensional drawing in time. This brings new aesthetic considerations to the table – for example, the seamless removal of old spirals from the screen and the ideal speed of animation.

The ultimate goal of the digital Double Meandering Algorithm is to create an animation that conveys the process of the drawing, rather than emphasizing the finished project. To recap, the original line and following line in a finished Double Meandering Line do not come across as independent entities. Instead, they tend to form a single double-lined line entity, perhaps even appearing as the outline of a rope. Yet, the separate natures of the original and following lines is an intrinsic part of the philosophy behind the Double Meandering Line. Hence, our primary objective is to convey the leading nature of the original line and the subordinate aspect of the following line in the animation.

As Bézier curves are traditionally drawn in line segments, we can easily draw partial curves simply by drawing more or fewer segments. A drawing-in-progress animation can thus be achieved simply by increasing the number of drawn segments from frame to frame. We employ the same principle in reverse to 'un-draw' the spiral, allowing for attractive and theme-appropriate erasure.

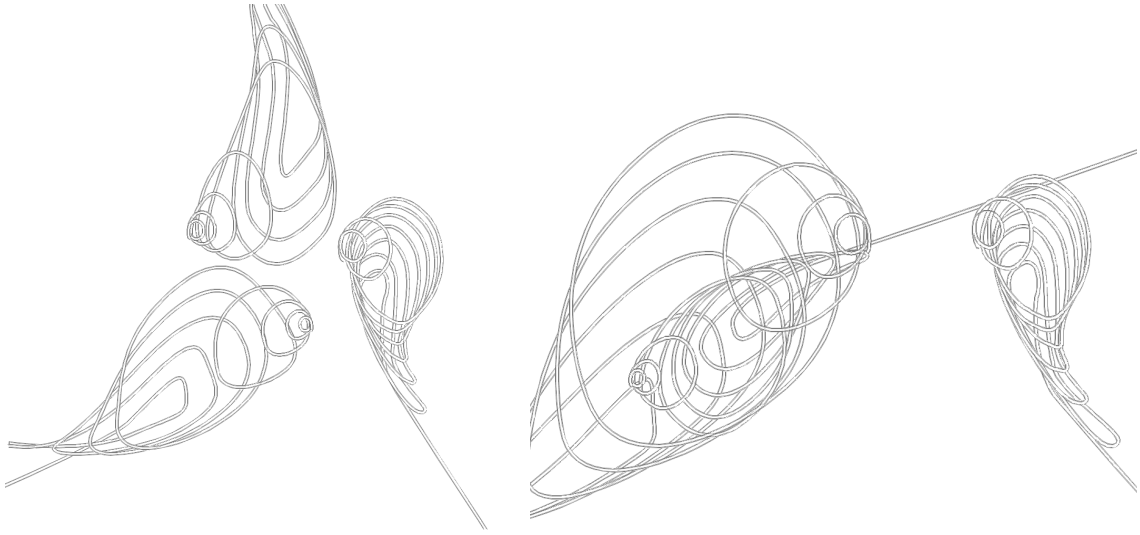

Figure 6: *Results of our Double Meandering Algorithm: complete drawings in our procedurally generated animation system.*

The original line is defined by the shape calculation step discussed in Section 4.2. The following line is simply drawn with an additional Bézier curve chain parallel to the original line, offset by 500% of the line width. To convey the separate and subordinate nature of the following line, the following line animation lags two full curves behind the first. This gives an orbiting appearance to the animation, as the two drawing lines are directly opposed to each other at all times during the drawing animation.

We must also deal with the illusory depth of the Double Meandering Line. Although our goal is a 2D drawing, we can easily use the benefit of a 3D rendering buffer and we do not need to calculate depth values in a complex manner. Instead we use orthographic projection and a slight offset into the screen for each iteration to automate occlusion. We replicate the overlapping curlicue loops in our algorithm simply by receding each line segment into the background by a single pixel. We also draw a background-coloured strip that accompanies the following line as it is being drawn. This strip lies between the two lines and covers the curlicues that recede further into the background, thus simulating the requisite empty space.

Since we designed the animation to run into perpetuity, we must remove the spirals from the screen in a pleasing way in order to showcase new ones. We achieve this by having the lines 'un-draw' themselves, erasing themselves from the screen in the same order as they were drawn. Alternate methods could include fading old spirals into the background as new spirals are drawn, or simply flushing the screen after all the spirals have finished drawing.

The current speed of animation is arbitrarily chosen and depends largely on the speed of the rendering engine. The Processing 3D API in Java takes approximately 2:41 minutes to complete the drawing and erasure of three spirals. While the rush of a faster progression is more aesthetically satisfying, the current speed is more accurate to the speed of the original pen-and-paper drawing and showcases the differences between each successive curlicue. Even if the ideal speed is faster than the current one, changing this element requires the animation to be modified to fit another engine.

5. Conclusions

The boundary between science and the arts is alternately judged to be either imaginary or impermeable, depending on the matter at hand – imaginary when the project is conceptualized, and impermeable when the objectives are not being reached as planned. Interdisciplinary projects are historically fraught with tension due to an unexpected inability for people with such different backgrounds to reconcile their differences. Ultimately, the success of an interdisciplinary project comes down to the ability and willingness of both parties to communicate. We believe that only by working together from the beginning on the digitization of the Double Meandering Line have we been able to successfully create the Double Meandering Algorithm.

The Double Meandering Algorithm is not 'canned chance' but a concatenation of singular aesthetic events. The pen-and-paper drawing is static. The algorithm is diachronic – the singular nature of each spiral becomes apparent in time. Increasing the amount of time one can spend with the operation is worthwhile. The result can be called a concatenation of singularity [Rau10], or a chain of distinct events that form a new thought. As aesthetic experiences go, watching the transition from one event to another is satisfying.

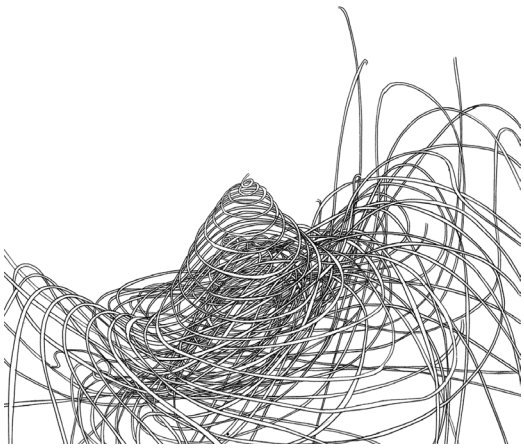

Figure 7: *An analog Double Meandering Line drawing by Lucy Pullen with a more complex base.*

In her later Double Meandering Line drawings, Pullen explores the 'meandering' element to a much greater extent by bringing the third dimension into play and abandoning the spiral structure altogether after a few initial loops. Our algorithm is completely unable to keep up with the scale of her vision, due to the huge discrepancy between the rigid rules of the digital drawings and the free-flowing, periodically enforced rules of the later analog drawings.

We are actively investigating what increases or decreases the sense of 3D space in the Double Meandering Algorithm. This is a compositional problem that is solved through the artist's conscious choices, and is not easy to quantify. One observer mentions that in order to establish the illusory 3D scope in which the drawings exist, the drawings appear to require at least two spirals with almost perpendicular spines to be on the screen at the same time. It also appears to be necessary to balance the sizes of all three spirals to consistently offer a sense of relative scale. This is one of the many cases where authenticity and randomization appear to be at odds with one another. In the future we will explore the parameter space of our algorithm and look forward to increasing our algorithm's flexibility to both mimic the same evolutionary steps experienced by Pullen's more recent drawings and to take steps that may not be easily explored in hand drawn Double Meandering Lines.

Acknowledgements

Thanks to the University of Victoria's Graphics group for all of their feedback and support. Thanks to Sarah Prusinowski, our editor-at-large. Thanks to our families for all their support. This material is based upon work supported by the NSERC Discovery Grant. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of NSERC.

References

[Gal03] GALANTER P.: What is generative art? Complexity theory as a context for art theory. *IN GA2003 – 6TH GENERATIVE ART CONFERENCE 2003* (2003). 2

[Gre05] GREENFIELD G.: On the origins of the term "computational aesthetics". In *Eurographics Workshop on Computational Aesthetics in Graphics, Visualization and Imaging* (2005), Neumann L., Sbert M., Gooch B., Purgathofer W., (Eds.). 2

[Hop00] HOPKINS D.: *After Modern Art 1945-2000*. Oxford University Press, 2000. 2

[KC03] KAPLAN M., COHEN E.: Computer generated celtic design. In *Proceedings of the 14th Eurographics workshop on Rendering* (Aire-la-Ville, Switzerland, Switzerland, 2003), EGRW '03, Eurographics Association, pp. 9–19. 2

[LOG06] LEE S., OLSEN S. C., GOOCH B.: Interactive 3d fluid jet painting. In *Proceedings of the 4th international symposium on Non-photorealistic animation and rendering* (New York, NY, USA, 2006), NPAR '06, ACM, pp. 97–104. 2

[Nes86] NESBIT M.: Ready-Made Originals: The Duchamp Model. *October 37* (1986), pp. 53–64. 2

[Rau10] RAUNIG G.: *A Thousand Machines*. Semiotext(e), 2010. 6

[Tar10] TARBELL J.: Complexification, 2010. 2

[XXK*06] XU S., XU Y., KANG S. B., SALESIN D. H., PAN Y., SHUM H.-Y.: Animating Chinese paintings through stroke-based decomposition. *ACM Trans. Graph. 25* (April 2006), 239–267. 2