Rodrigo Aguirre

Logics

An Exploration of Digital Drawing Logics in Code

Introduction

"Art and science have their meeting point in method". Edward G. Bulwer-Lytton.

Drawing is traditionally seen as the act of creating an image through a relational means between the artist and the medium. In today's digital era, the artist is subjective to human or machinic protocols. If the act of drawing is unique to humans, what does it mean to automate the procedure of drawing? Does it take away a certain unpredictable quality? The human touch, as they call it. Or can a machine portray its own sense of unpredictability? A set of randomness. A noise. The Scripts' drawing machines are programmed to create images using a 0.1 point, black ink pen on paper. They function using input data when given a relative time and produce a complete drawing. What does it mean to then reproduce the automated drawing through a digital simulation? A code that creates a similar output image? Can one reproduce the unpredictability of random machine input through computational calculations?

The digital drawing machine (DDM) addresses the idea of the replica. However, rather than replicating an image, it replicates a set of behaviors. Ergo, the digitally simulated drawings are an expression of behavioural settings produced on a digital canvas. The DDM, as such, consists of two programmable procedures; one at the hardware level, the other through software. Initially the physical model is created, through experimentation with forces, flows of energy and the possible variety of brushstrokes that can emerge. Then the procedure is mimicked, that is the act of drawing through code on a digital canvas.

Process of Replication

Reproducing behavior through code necessitates a set of rules which determine the response of the algorithm based on the specifics of a situation. In other words, a program set of conditions need to be established that alter the output. To isolate specific moments, one must observe a certain movement, understand that there is a force acting on the object,

that is the pen, and realize how the object is reacting. The process begins with an assumption, and is followed by a repetitive trial and error stage. The assumption may be correct, or may need continuous experimentation. Realizing adaptation includes the addition of rules if missing discipline, or their simplification where complex. The process continues until the most accurate iteration is realised.

A set of **parameters** are defined to translate the performance from physical to digital:

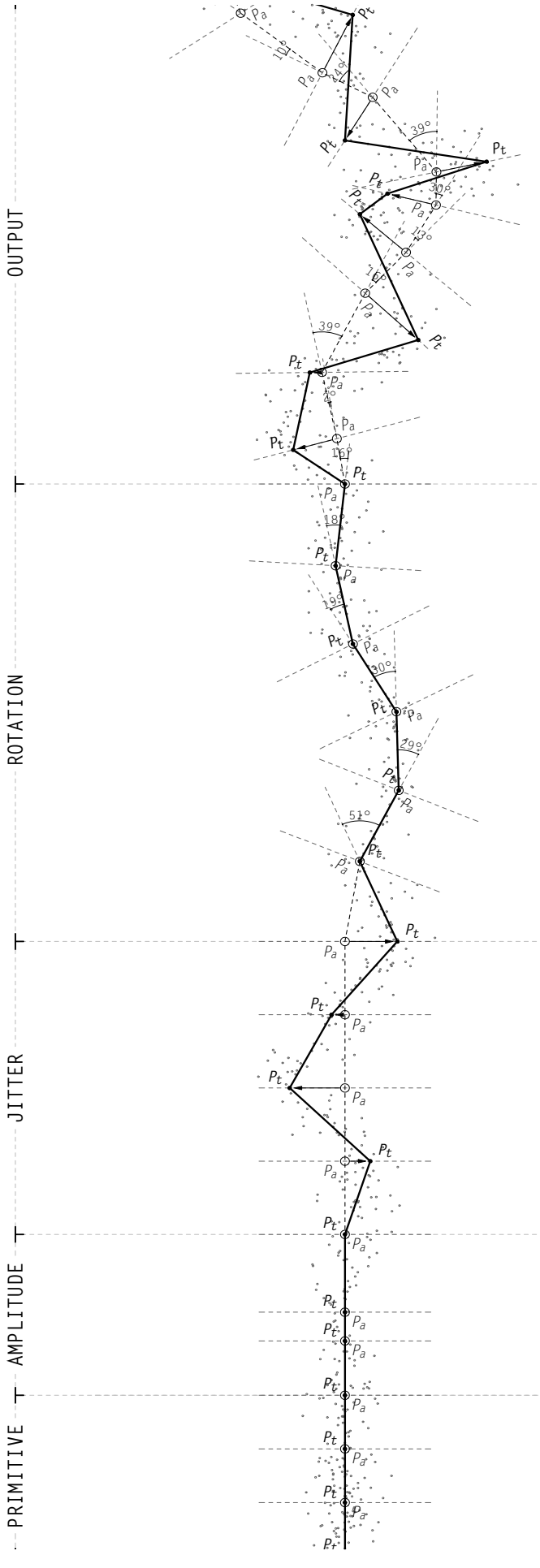
- Behavior = driving forces and global constraints
- Pen = a point elaborated by a coordinate system
- Time = iterations (repetition loops)
- Variable = input data, in this case controlled but random values

In order to simulate, or replicate, the **behavior** of the machine, it should first be analyzed. The behavior responds to global constraints and driving forces. The global constraints will 1// dictate the type of movement the pen will make, or 2// define the general movement of the object, which is the pen. The driving force will act, or move, within this constraint and consequently affect the stroke, or type of trace, left by the pen.

The pen is a coordinate system, a moving point in space. It can be determined whether the trace left by the pen is a scattered series of points, or a continuous trail, read as a line. Over a period of iterations, a driving force moves the point. The constraints limit the movement of the point in three dimensions, for instance it is bound to euclidean operations: translation, rotation and orientation.

Time is inevitably a parameter to consider as the machine will produce a drawing over a finite amount of iterations. For the amount of time that the machine runs, the pen is affected in each of its movements, this is in understanding that with every new condition that the machine experiences, it will consequently alter the next step. Periodicity can translate to computation with iterative functions such as loops.

Variables are the input data for the parameters. By controlling the parameters it is possible to create a catalog of performances, or simulations, and extrapolate different sets of

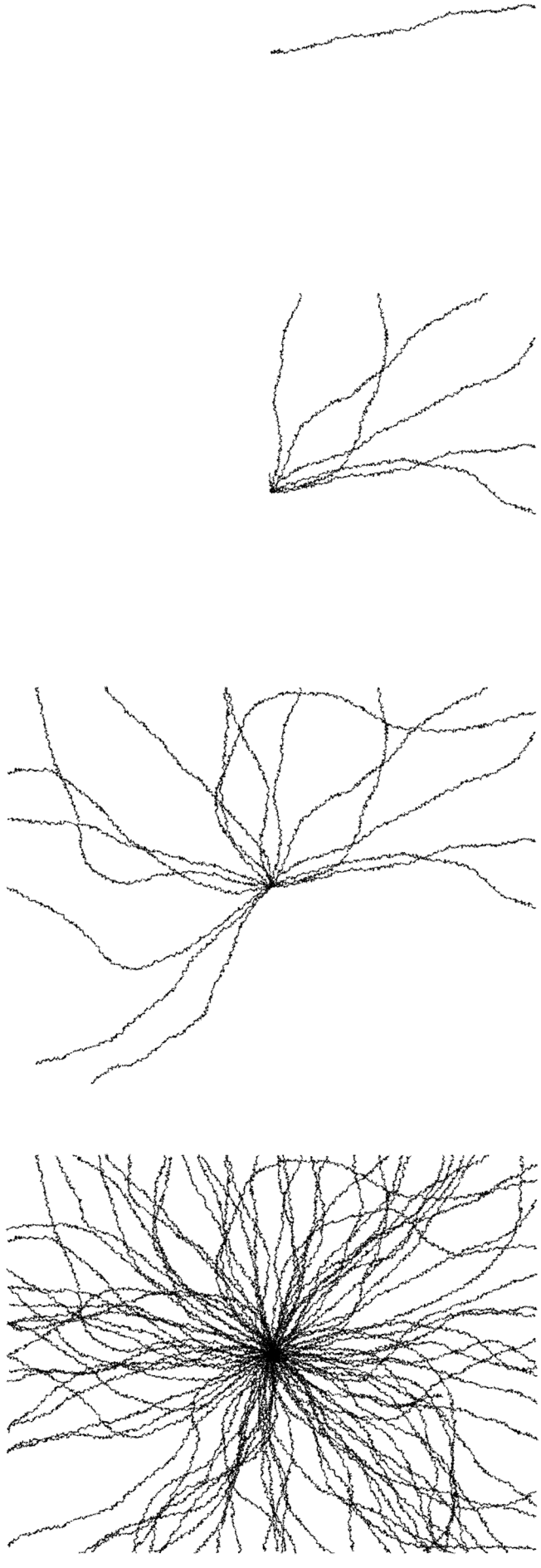


Trail of the hidden line, dissection of parameters.

variables. The output shows the many possibilities that can result from these combinations. The input data can be programmed, subject to sensors, or even a random factor. This analysis helps to debunk the conditions of the machine into the digital canvas.
Another question that should be considered is how imperfections in marks can be replicated by the DDM. The answer? By adding noise rather than attempting to accurately predict the result of each vibration, **the random factor** is introduced as an estimated set of input values to achieve a similar output quality as that found in drawings by hand. In other words, the stroke created by the pen has an indeterminate quality unique to each machine. By controlling the limits of the random factor's domain, it is possible to create undetermined variables within the given range; allowing a controlled freedom. Imagine a point placed randomly inside the space of a box with the ability to control the dimension of the box, but not the location of the point.

Case study
Machine #1: Spider

A pen is placed in the center of the machine, held vertically by a rigid, triangular structure.This drawing tool hosts four small motors, one on each leg of the triangle structure and the fourth on the pen itself. This system transmits vibration to the central point that is the position of the pen. Due to the vibration of the motors the trace left behind is a set of points jittering around its main axe.
The machine is placed in the center of the page and while the program is running, the motors vibrate, moving the object and consequently leaving a jittered trail. This movement continues until the machine reaches the limits of the paper, when the boundary is hit, it is placed in the center once more and the process is repeated over a period of time. As the machine motors vibrate on each leg they create two movements: rotation and translation. By controlling the vibrations on each leg, the drawing tool structure is allowed to move in any direction.
To start the computational process, the machine structure is simplified to a segment. The core of the script runs in **loops**. Starting the loop, the line rotates around a given point



The evolution of time from 200 to 20000 iterations

along a curve. After this operation takes place, the new rotated line moves perpendicular to the direction of the first segment, that is the curve, resulting in a side movement. Both the angle of rotation and the amplitude of translation are controlled with random values inside the limits of a controlled domain. The resulting position of the line is then evaluated to find a point within its domain, the consequence of this action will create the jittered path. After this process, the point location is recorded to observe if the point is inside the drawing bound. If the point is out of bounds, the line returns to the origin. The line position is then saved for the next iteration, this marks the end of the loop and the process is repeated.

Defined Parameters

- Behavior = Jittered trail
- Global constraint = rigid, triangular structure
- Driving force = motor vibration
- Pen = scattered point trail
- Time = programmed interval
- Variable = controlled random domains

Catalog of Explorations

The catalog is a series of 9 drawings that explore the range of resulting possibilities when incrementally manipulating each of the **defined parameters**. There are three categories to the catalog, each containing four performances. Each catalog manipulates one set of parameters at a time while keeping the rest constant in order to observe the variety of outcomes.
The first catalog discusses the way the **behavior** is being translated into code by defining each parameter. The **hidden line** is the order behind the drawing, the lines that define the drawing but will never show on the paper, similar to the axe lines found in a structural plan. By employing different types of movements and ranges of variables to the forces and constraints, the journeying process of the hidden line is evolved.
The second catalog is about **time** in terms of iterations. The four drawings in this series show a variance of outcomes when enlarging the number of looping repetitions of the script. The longer the script is running, the more depth and detail the drawing attains. In the first drawing of the series, a minimum of 200 loops is chosen to show a single line. The second drawing of the catalog is 1100 iterations, showing the lines filling the page. This is approximately where the pattern begins to repeat itself. The third drawing with 9000 loops shows the possibility of evolution. Finally, the drawing with 20 000 loops fills the page, and the process can continue indefinitely, nethertheless it will require longer processing time.
The third catalog explores varying possibilities of the drawing when manipulating all of the **defined parameters** in correlation to the machine behavior. This means that this catalog shows a wide range of possibilities when playing with the code. Contrary to the second catalog, in these four examples time, or the number of iterations, is fixed while the other parameters such as the jittering of the pen and the random factor are differentiated.

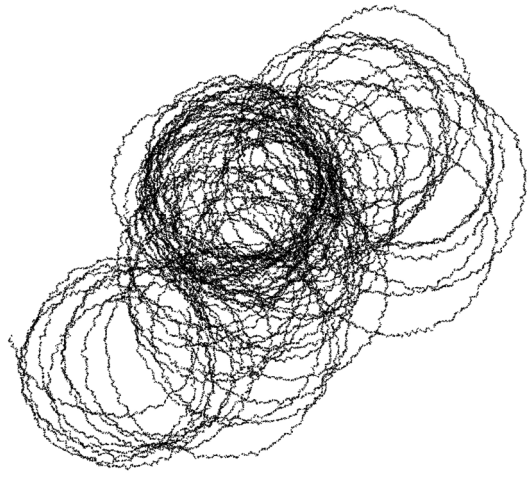
Conclusion

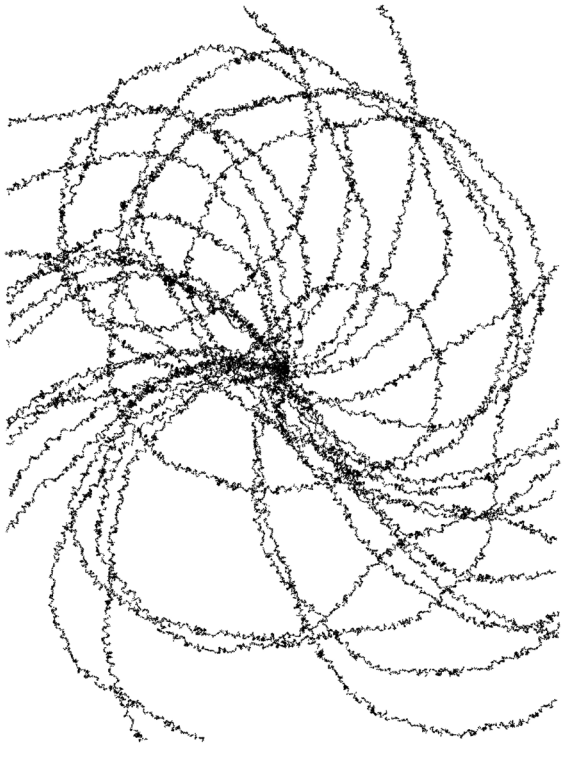
In attempt to determine the machine behavior, one must control the output DDM performance. This can be done by manipulating the following parameters: first, the code inputs, second, the iterations. When generating a computational drawing it is less about predicting the final result, and rather more focussed on the **characteristics of the drawing**. The additional rules act as constraints, resulting in consequential behaviors similar to the physical machine.

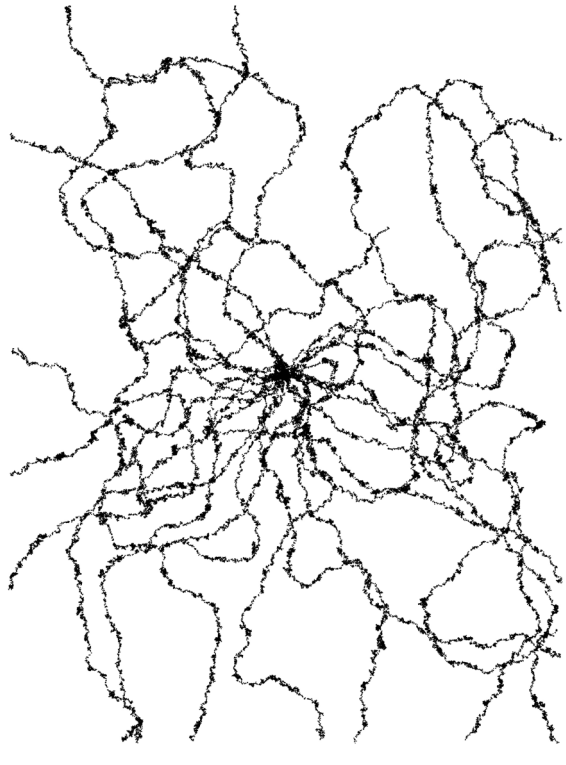
"The power of recursion evidently lies in the possibility of defining an infinite set of objects by a finite statement. In the same manner, an infinite number of computations can be described by a finite recursive program, even if this program contains no explicit repetitions." Wirth, Niklaus (1976)

When drawing with recursive programming, advancements of code allow for many iterations to be produced in less time. There is a direct relation between the code produced and the machine function. By exploiting the potentials of the code one can achieve a variety of outcomes in terms of design. With the manipulation of parameters and interchangeable variables in a code, the drawing evolves.

Having established the logic in which is needed to recreate, or replicate the behavior of the drawing machines, the question is resolved: can one recreate the unpredictability of the machine itself? The drawing machine functions with a human intervention, as do the digital drawings, each is a programmed set of behaviors, able to find their direction via code. Once the logics have been produced and reproduced and the digital canvas is simulating the replica of the physical, there is slight distinguishing between the two outputs. Humans have the capability to create such behaviors and to replicate the behaviors into performing once again on their own. The Digital Drawing Machine challenges the limitlessness of the relation between art and technology.







Manipulating the variables of behavior; global constrains and driving force.