GA2: a Programming Environment for Abstract Generative Fine Art

Philip Galanter, BA, MFA
Interactive Telecommunications Program, New York University, New York, USA.
e-mail: galanter@nyu.edu.

Abstract

Fine artists looking to use computers to create generative works, especially those artists inclined towards abstraction, often face an uncomfortable choice in the selection of software tools. On the one hand there are a number of commercial and shareware programs available which implement a few techniques in an easy to use GUI environment. Unfortunately such programs often impose a certain look or style and are not terribly versatile or expressive. The other choice seems to be writing code from scratch, in a language such as c or Java. This can be very time consuming as every new work seems to demand a new program, and the artist's ability to write code can seldom keep pace with his ability to imagine new visual ideas.

This paper describes a software system created by the author called GA2 that has been implemented in the Matlab software environment. By layering GA2 over Matlab the artist can take advantage of a very mature programming environment which includes extensive mathematical libraries, simple graphics routines, GUI construction tools, built-in help facilities, and command line, batch mode, and GUI modes of interaction. In addition, GA2 is very portable and can run on Macintosh, Windows, and Unix systems with almost no incremental effort for multi-platform support.

GA2 is a work in progress and an extension of the completed GA1 environment. It is medium independent, and can be used for all manner of image, animation, and sound production. GA1 includes a complete set of genetic algorithm operations for breeding families of graphical marks, a database function for managing and recalling various genes, a set of statistical operations for creating various distributions of marks on a canvas or animation frame, a unique Markov-chain-like operator for generating families of visually similar lines or paths, and a complete L-System implementation. GA2 extends GA1 by adding more generative techniques such as tiling and symmetry operations, Thom's cusp catastrophe, and mechanisms inspired

by complexity science notions such as cellular automata, fractals, artificial life, and chaos. All of these techniques are encapsulated in genetic representations.

1. GA1 Goals and Advantages

The first software system I created for creating generative art I named GA1. My intent was not to create a software tool for general distribution, but rather to create something I could use for my own artwork. At the same time I wanted GA1 to provide a general extensible environment that would yield rewards across a number of projects in a number of forms and styles. Having said that, GA1 (and now GA2) is generally useful, and I hope to make it available to some student artists at NYU in the fall semester of 2001.

For the sake of economy, one trade off I willingly made was to not require that GA1 produce its results in real time. Like most commercial computer animation programs, GA1 executes in phases, the last one being a rendering phase where a single minute of graphics or sound may take a number of minutes, or even hours, to compute. GA1 is also not intended to be the engine for interactive works. By lifting any real time requirements GA1 can be programmed at a high level, and software rigor and elegance do not have to take a back seat to execution time.

GA1 is not a single program, but rather a system of related software modules. GA1 includes subroutine libraries, commands that can be used interactively, point-and-click GUI tools, and batch oriented compute-intensive programs. Because GA1 is more like a soup than a monolithic system, any technical overview will be a bit disorienting at first. Perhaps the most important aspect of GA1 is that it presents a modular paradigm that can embrace a wide variety of techniques and algorithms, including (I hope!) many that have not yet been invented.

Because GA1 is implemented in Matlab, it is portable across a number of hardware and operating system platforms, including Mac OS, Windows 95/98/NT, and Unix. For example, initial work can be done on an inexpensive Mac, and later compute-intensive batch jobs to create a final animation can be run on a large Silicon Graphics “number cruncher”.

GA1 is not intended to be an all-in-one package. By design it is intended to be used with and leverage off of commercially available software packages, taking advantage of them where they are useful, and optimizing the artists time by programming only what is otherwise needed.

GA1 is not a static environment, but rather an evolving system that allows the artist-programmer to build, tweak, and modify tools as needed on the fly.

For purposes of illustration very simple examples are presented here rather than rich final compositions. GA1 allows the artist to assemble any combination of these techniques to create both stills and animations that are complex and subtle.

Viewed as a design tool GA1 offers a number of advantages:

database function - Various styles, line thickness, color combinations, sonic characteristics, motion tendencies, and so on are encapsulated genes. Sets of genes are concatenated together to form chromosomes, and a single chromosome can be expressed multiple times to make numerous marks or sounds which exhibit variation within the same aesthetic look or feel. GA1 includes utilities for the creation, management, cataloging, and recall of these chromosomes. The point here is that once I have developed a given look, it's in the database and it can be reused later at any time.

Variation generation - Given a library of chromosomes, i.e. individuals within a given species, I can breed and mutate given chromosomes to create stylistic variations. So if something looks or sounds almost but not quite right, I can generate a number of variations to select from and perhaps breed again. For a recent series of large stills presented as transparencies in lightboxes, I used 3 species of marks, with about 200 individuals to choose from within each species.

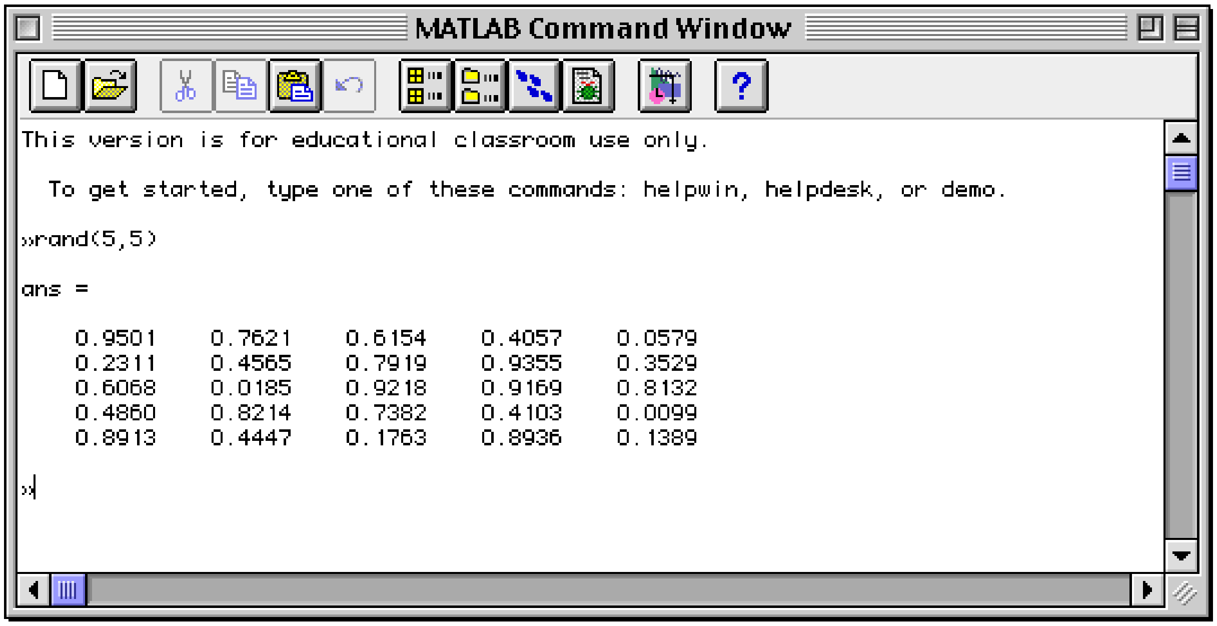
Non-repeating patterns – GA1 can fill an arbitrary planar space of any shape with a given design, look, or style without using any tiles or other form of repetition.

A nonlinear production response - most computer based design tools are somewhat similar to manual tools in that for a single physical gesture you create a single mark. Whether using Photoshop, Painter, or Illustrator the digital product is still in a sense handmade. The creation of new artwork is somewhat more productive, but is still very time consuming manual labor. GA1 can yield a nonlinear return on the artist's effort. Once a look or style is developed, with very little incremental effort the designer can decorate any number of objects, surfaces, or installations.

A massively parallel architecture – Much of GA1 is compute intensive, but structured in a highly parallel fashion. While the parallel structure is not currently exploited the promise remains that future hardware with multiple processors will yield a nearly linear improvement in execution time.

1.1 Matlab as a Programming Environment for Artists

Matlab is a programming environment with a number of features that recommend it for use by generative artists. While it can be used for compute intensive "batch" processing, any part of the Matlab language can be used from the command window interactively. As users write their own routines they can be bundled into so called "toolboxes" so they are fully integrated into the interactive environment. Matlab also has an extensive help system, and user authored help can be seamlessly integrated into that aspect of the system as well. In short, Matlab can be highly customized to create a digital workspace closely fitted to the user's needs.



```
This version is for educational classroom use only.

  To get started, type one of these commands: helpwin, helpdesk, or demo.

»rand(5,5)

ans =

    0.9501    0.7621    0.6154    0.4057    0.0579
    0.2311    0.4565    0.7919    0.9355    0.3529
    0.6068    0.0185    0.9218    0.9169    0.8132
    0.4860    0.8214    0.7382    0.4103    0.0099
    0.8913    0.4447    0.1763    0.8936    0.1389

»
```

Figure 1 – The MATLAB command line interface.

In the example shown here a random number function has been called to create a 5 by 5 matrix of random numbers between 0 and 1.

Matlab operators are overloaded to allow array operations without having to tediously code explicit loops. In addition Matlab has a number of mathematical libraries, freeing the programmer from having to reinvent common functions. In addition the Matlab routines are often (but not always!) faster than the routines most users would code at a lower level.

Matlab also provides built-in engineering graphics tools, and a set of Graphic User Interface (GUI) routines for the creation of point-and-click controls. Matlab's graphics are not adequate for formal fine art work, but in GA1 Matlab graphics and GUI tools are used for the preparation of control surfaces, paths, fields, and other spatial data types.

2. GA1 Overview

Generative art systems typically produce works given an initial set of conditions and a mechanism by which those initial conditions are set into motion to interact and develop. This interaction produces a result that is, at least to some extent, surprising even to the artist. The degree to which the artist gives up control or shapes the result is, in itself, a choice the artist makes. Some systems, by design, allow for little artistic control, while others are designed to be artist driven.

The model used by GA1 allows for a great deal of flexibility with regards to artistic control versus machine autonomy. In addition GA1 allows for the production of both 2D still and motion picture with sound works...and could be extended to include other forms such as sculpture.

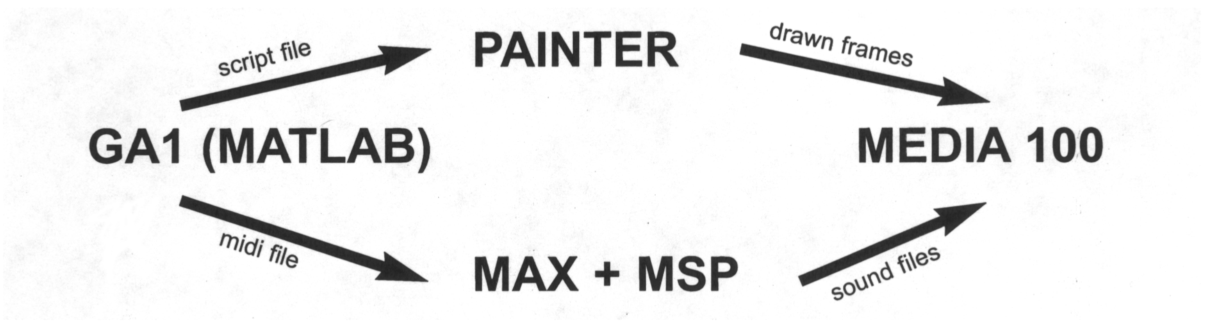


Figure 2 – GA1 data flow as used to create a video with sound.

Most of the programming used to implement GA1 has been done in an engineering software environment called Matlab. In this example GA1 (via Matlab) generates two files, a script file for the Fractal Design (now Corel) Painter application, and a Midi file for the MAX music application with the real time audio MSP extension. In a sense the composition is "finished" when GA1 creates these 2 files. The next steps are fairly mechanical. Painter interprets the script file generating a large number of video-sized frames. A MAX+MSP patch I've created executes the Midi file synthesizing a sound file. The sound files and frames are then matched together on a Media 100 video editing system for the final picture and sound.

For the most part GA1 is independent of the final rendering mechanism. It would be a fairly straight forward matter to add device drivers for painting machines, numerical control milling or stereolithography machines, or other industrial rapid prototyping tools for the creation of objects rather than pixel based images.

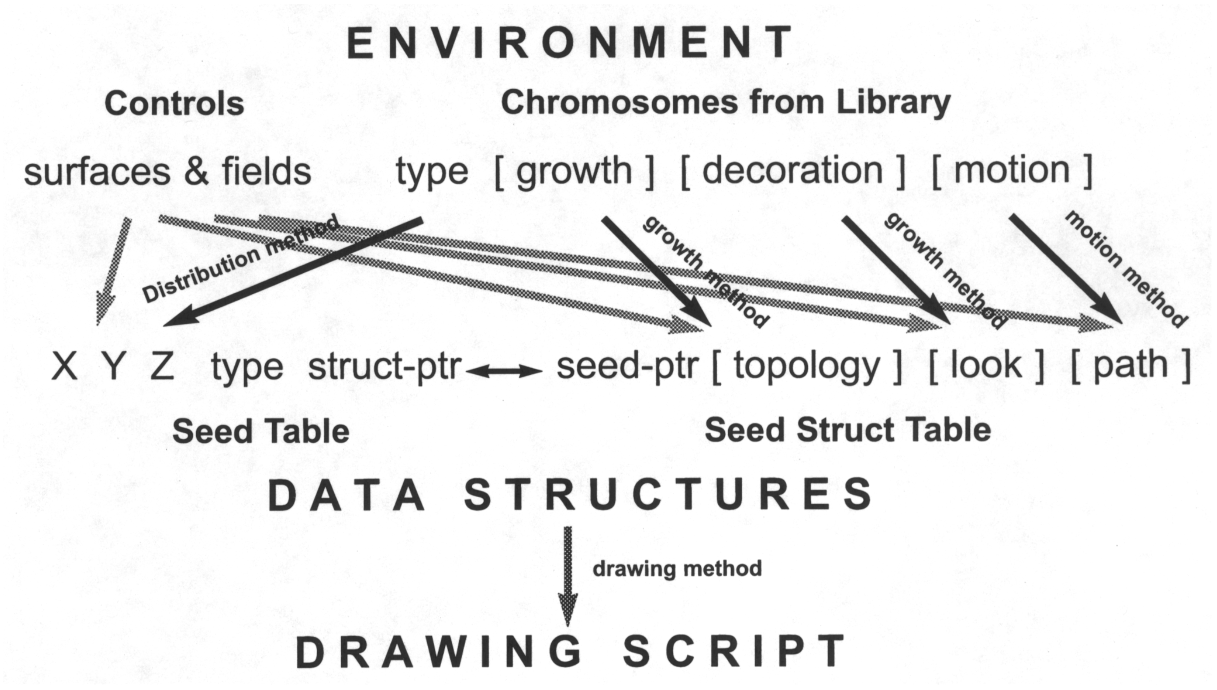


Figure 3 – GA1 internals diagram. Chromosomes and controls interact and are expressed as complex data structures. These, in turn, via drawing methods and device drivers create scripts for rendering devices or software.

GA1 as implemented in Matlab is summarized in the above diagram. In the first step a "seed" for each mark or object is placed on the (virtual) canvas using one of many distribution methods. Each seed corresponds to an (x,y) canvas location along with a z coordinate which indicates the back-to-front order in which the marks are drawn. Each seed has an entry in the Seed Table, which can later be used to quickly sort and select seeds by type and location. (Type here is a proxy for species id number, an individual's id number, or an arbitrary grouping number specific to the piece.) The Seed Table is linked by pointers with a much larger Seed Structure Table that is empty when the seed is first distributed.

In this system each seed and corresponding mark or object drawn has a chromosome with genes which determine how the mark is grown, how it is decorated, and in a motion picture, how it moves about. Related growth and motion methods are typically specific to a given species, but often call a library of shared utility and custom math functions. Along with chromosomes the environment also includes surfaces and fields which control and modify the innate behavior of a seed based on its (x,y) location. The environment provides input to the growth and motion methods, the results of which are stored as complex data structures that completely describe the final topology, look, and motion path for each seeds mark.

(Note: In the context of this paper the term “topology” is used in an informal way to refer to a sort of stick figure approximation of the final form without any decoration such as color, paint texture, or the like. It is not a reference to the mathematical notion where malleable shapes retain abstract relationships.)

Once the seed structures are completed, a species and medium specific drawing method is used to render each mark. By providing alternate drawing methods and drivers, devices such as painting and milling machines could be used rather than the virtual painting machine offered by running Fractal Design Painter with scripts.

3. GA1 Functions and Operations

The completed GA1 system includes a number of functions and operations that are described here.

3.1 The Chromosome Library and Breeding

Each species can have a number of individuals, and each of these individuals will have a unique chromosome in that species’ format. New individuals can be created by taking a single individuals chromosome and introducing mutations, or by taking the chromosomes from two individuals and breeding them yielding new chromosomes via crossover recombination.

It is also possible to extract specific genes from a given individual from one species and to splice it into the chromosome for an individual of a different species. There is, however, the following restriction.

The genetic data structures used in GA1 are not pure bit strings, as used in classical genetic algorithm classifier programs, but rather are higher level data structures specific to a given function. Thus, for example, a color gene from a given individual can only be spliced to replace a color gene in another individual. GA1 has both species specific genes and standardized genes common across all species, such as those for color, absolute length, relative width, and so on. For the purposes of gene splicing only the standardized genes can be spliced from one individual to another. Within a species, however, the chromosome can be crossed over, mutated, or spliced at any gene site.

3.2 Surfaces and Fields

Surfaces and fields are controls the artist can interactively create with GA1 and then use in compositions. Multiple surfaces and fields can cover the virtual canvas, and each can influence an aspect of the way the seed at given point grows. This is not unlike the way that nutrient concentration in the soil (a GA1 surface) or the prevailing wind patterns (a GA1 field) will ultimately influence the form a plant will take. The growth and motion methods for each seed sample the local surface and field values for use as the given genes are expressed from genotype to phenotype.

Depending on the species, surfaces and fields can be used to modulate all aspects of color, site, shape, orientation, motion speed, audio pitch or timbre, growth or motion direction, and so on. The chromosome from a single individual can thus be expressed in a number of ways across the canvas. This can be under the direct control of the artist by his manually designing the control surfaces and fields, or such variation can emerge on its own by having GA1 dynamically create control surfaces and fields during execution.

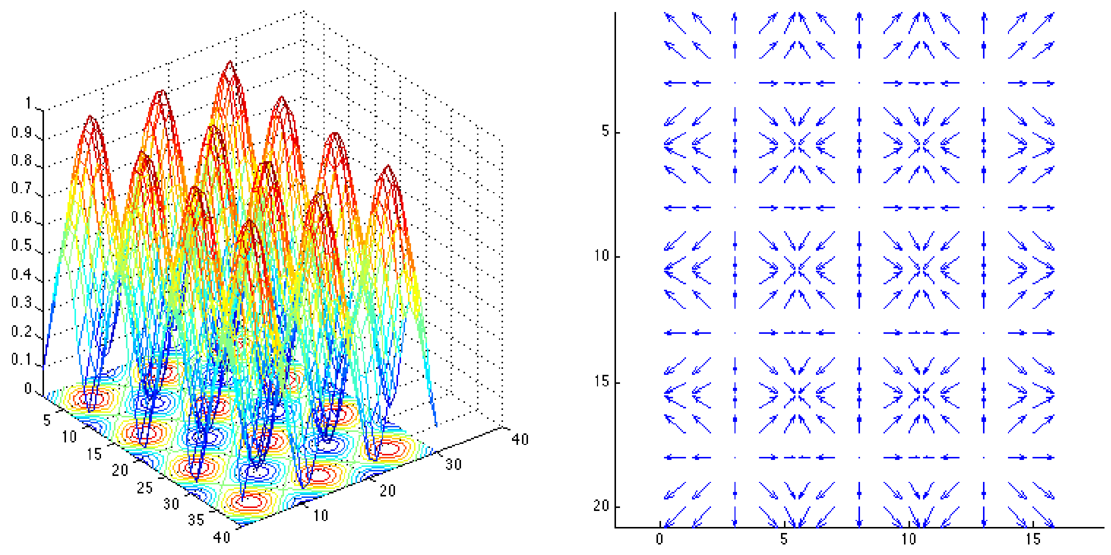


Figure 4 – A control surface and the same control surface converted to a field.

The GUI control panel shown at the bottom of the page allows the artist to sculpt a 3D surface with arbitrary slopes, peaks, valleys, etc. Usually the surfaces are depicted using an elevation map such as the one shown. When displayed in color red areas are the highest altitudes, and blue the lowest. The surface can be thought of as assigning a value for every point on the canvas, some high (red) and some low (blue).

A field, such as the one shown here, assigns 2 values for every point on the canvas, both a magnitude and a direction. The artist can create a field from a surface by calculating the surface gradient via the GUI control panel. An intuitive way to think about this is if a marble was placed on one of the red peaks below it would roll into one of the blue valleys. The corresponding field shows arrows (force vectors) leading away from the red peaks and into the blue valleys.

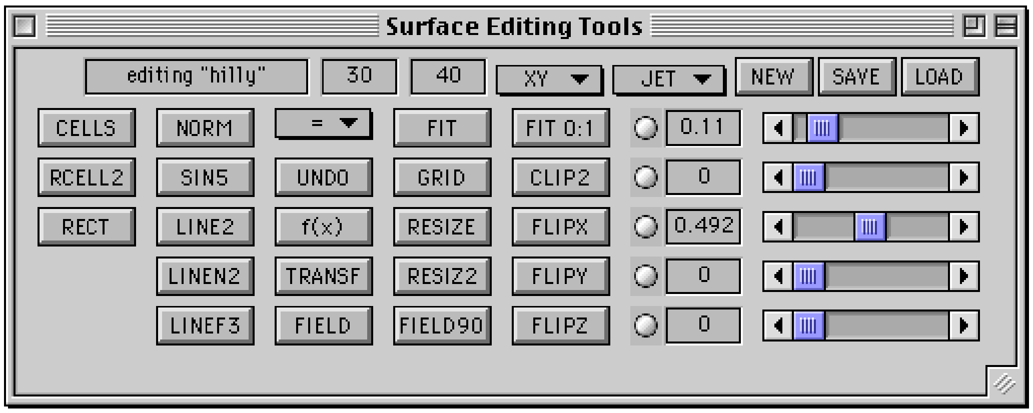


Figure 5 – The control panel used to create and sculpt control surfaces and fields.

3.3 Seed Distribution

One of the first steps in any GA1 composition is the distribution of seeds. Seeds simply indicate the spot on the canvas where a particular kind of mark will be made or at least started.

One useful method is the random distribution of seeds within a border. The diagrams shown here are from tests made during the early development of GA1. As can be seen in the plot, when seeds are distributed within a circle using uniform random numbers, the distribution appears “lumpy” rather than “smooth”.

When using GA1 seeds are often first randomly distributed, and then iteratively moved about to achieve an even distribution as defined by a stopping function. In this case the distances between each seed and its nearest neighbors are measured. Seeds that are “too far away” are moved a bit closer, and seeds that are “too close” are moved a bit further away. After many cycles of adjustment the variation in distances settles down to a constant within a predefined tolerance. In this case variation was measured as the standard deviation of the distance between seeds

Having thus satisfied the stopping function, the newer “smoother” distribution of seeds is used in subsequent steps.

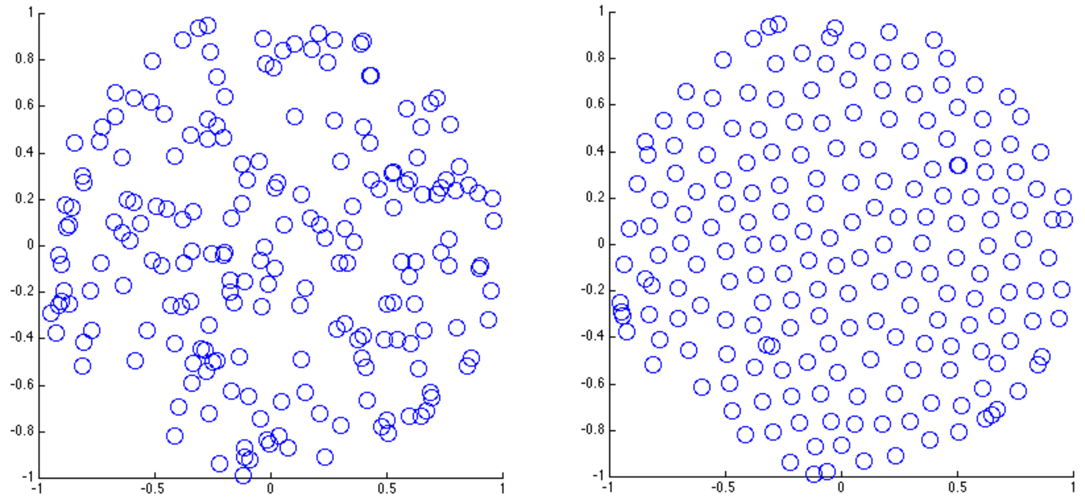


Figure 6 – A uniform random distribution of seeds before and after smoothing.

Seeds need not, however, be uniformly distributed. A control surface can be used as a density function to define areas where the seeds are placed in greater or smaller numbers.

3.4 Path Generation

Paths are a basic data type used where there is a need to create brush strokes, motion paths, or any other aspect described by a curved or straight line. A given path gene can be expressed multiple times to create an arbitrary number of paths that are unique yet similar in style.

Path genes for any species can be developed using the GA1 path tool as shown on this page. It allows the artist to generate path genes by trial and error. The genotype is set to random values within a set of constraints defined by the artist. The gene is then expressed multiple times, each initially starting in the same direction. Where the phenotype (the visible form) shows promise the corresponding gene can be added to the gene pool of the given species.

GA1 includes a proprietary mechanism for encoding as a gene tendencies which result in families of similar looking paths. A single routine executes all of the path genes for all species. If two individuals are bred and crossover occurs at the site of the path gene, the resulting genes will generate paths that show similarities to both parents.

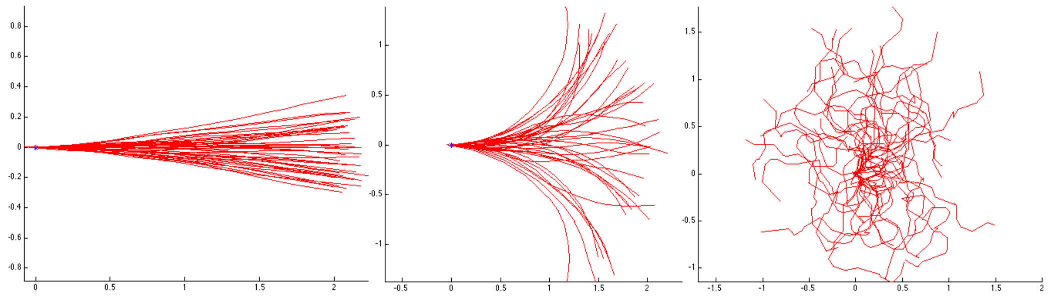


Figure 7 – 3 path genes expressed multiple times in each plot demonstrating variation.

In addition, path genes can be expressed in an environment where there are barriers. The resulting path will avoid the barriers by executing turns that are consistent with its typical overall shape and behavior.

Finally path genes can be expressed in the presence of a field. Just as the prevalent yearly wind conditions will, over time, warp the way a tree grows, fields can direct the way a path develops without robbing it of its intrinsic shape or character.

While not shown here, path genes can encapsulate fairly complex behaviors. For example, a given gene could give rise to a repertoire of behaviors such as a mix of small clockwise loops, large counter-clockwise loops, and long straight connecting strokes.

3.5 L-Systems

L-Systems are a family of grammar based methods which can be used to simulate and generate a wide variety of natural branching structures. As a part of GA1 L-Systems are not the primary center of attention, but simply one of a number of modules or alternatives for generating form. Also the use of L-Systems in GA1 isn't specifically intended for the creation of representational art, such as drawings of trees and other plants, but rather the use of natural branching forms reinterpreted for use in abstract art.

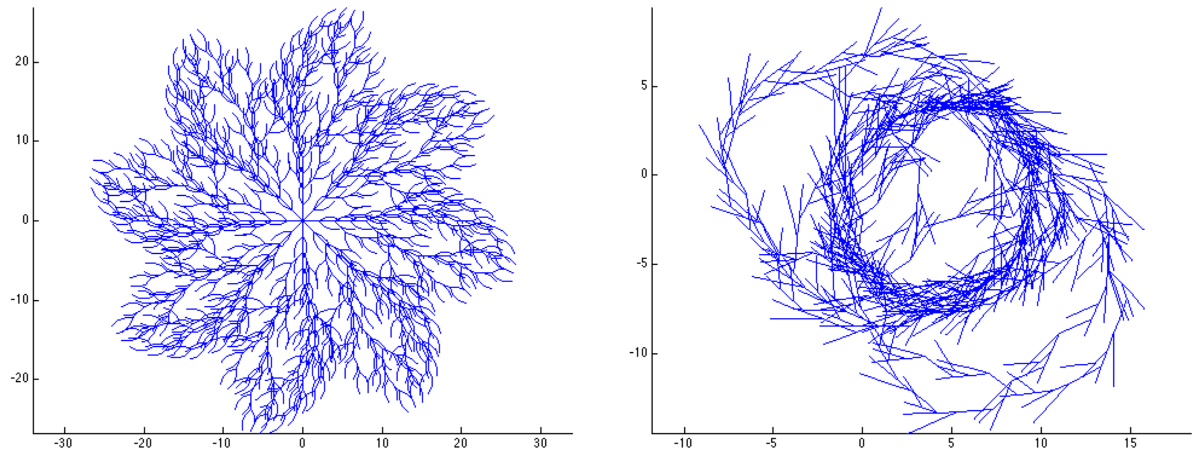


Figure 8 – Two L-System genes developed and expressed in the GA2 environment

GA1 includes a robust L-System engine, and a method for encoding L-System grammar in a genetic structure. GA1 L-Systems include stochastic and contextual functions. Parametric L-Systems will be supported in GA2.

Both L-System and path genes are growth related genes which only contribute topology. A full rendering also requires decoration genes that contribute a look, and usually some additional smoothing which eliminates most of the hard angles.

4. GA2 Improvements and Additions

GA2 is a software project currently underway and an extension of the intent and mechanisms of GA1. Following are a number of improvements and additions either under consideration or in active development. A primary thrust in the development of GA2 is the incorporation and application of methods from the realm of complexity science.

4.1 Adoption of the Open Source Platforms Octave and the GIMP

While the use of commercial software as part of the GA1 environment yielded quick results and minimized the amount of programming required before actual artwork could be created, some of the problems of using commercial software were also encountered. Support for Matlab on the Macintosh has been dropped. This is unfortunate for computer artists, as the Mac is the leading platform for many related forms of media production such as music and video production. Its unclear whether Matlab support will return with the advent of the Macintosh OS X.

Similarly Painter has had its ups and downs in terms of vendor support. In addition, Painter's scripting capabilities have been problematic when working with large files, such as some recent 12,000 by 9,000 pixel resolution images.

The open source community now offers viable alternatives to both, at least relative to GA2 requirements. Octave is a sort of Matlab clone in terms of its programming language, and may allow for an easy port of GA2. However, Octave's graphics support is acknowledged to be inferior to Matlab.

GIMP (or "the GIMP") is an open source image-editing package that includes paint tools, and can be used in a manner similar to Photoshop or Painter. GIMP can also be used via scripts and can run as an image-rendering server. This would allow GA2 to be used for web based art projects involving the dynamic creation of generative art.

4.2 Iterative Processing Allowing Computer Vision

GA1 is a one way pipeline where the generative aspects are executed, and then a rendering system creates the final output. Methods are being explored to allow GA2 to iteratively generate elements, render them, read in the resulting image, use computer vision techniques to (for example) identify areas in the image for further work, generate additional elements, render them, and so on. This may be done by connecting either Octave or Matlab to GIMP via Perl scripts. Its unlikely this could be done using the current version of Painter.

Such a mechanism would also allow GA2 to process real world photography or video footage. One can imagine, for example, artwork created via an autonomous generative art rotoscoping system.

4.3 Tiling and Symmetry Procedures

A number of routines are in development to support symmetry operations about a point, about a line, or filling a plane, as are a number of tiling procedures. Along with the obvious uses for tiles, one can use tiles as another method to distribute seeds. In addition, rule based systems can process regular tiles and combine them to form complex areas or contours. Finally exotic tiling patterns can be used to explore cellular automata which are not in the typical square grid configuration.

4.4 Cellular Automata

Control surfaces in GA1 and GA2 are simply arrays of floating point values that approximate a surface but are not continuous. (When values for a specific point on the surface are needed, however, they are calculated via Gaussian interpolation, and thus the surface can be used as functionally continuous). Since surfaces are simply rectangular arrays they can also be used as the grid for cellular automata. A set of generic cellular automata routines will allow artworks, especially animations, to exhibit various forms of emergent behavior. For example reaction-diffusion processes could be simulated which would then act as a control surface to create the kind of organic looking stripes and spots one sees in seashells, fish, and other animals.

4.5 Thom’s Cusp Catastrophe

While catastrophe theory has not lived up to its initial promise as a broad tool for model building, it remains a useful artistic tool for situations where one wants a generative behavior which exhibits bifurcation, hysteresis, and sensitivity to initial conditions. Basic functionality of this kind would be most useful in time based art, but could also be useful in creating processes which exhibit emergent behaviors which ultimately result in a single image.

4.6 Fractal Noise Generation and Chaotic Simulation

One of the oldest generative art strategies is the use of chance operations. The GA2 implementation of additional random number generators exhibiting non-uniform distributions or autocorrelation, such as 1/f noise, will be fairly straightforward and quite useful. The accurate simulation of truly chaotic dynamical systems is, of course, an exercise in frustration. But as an area of active research for the computer artist, the development of software systems that are “chaotic enough” is certainly worth exploring. The “feel” of chaos in a time based work, whether visual or sonic, is very fertile ground for the artist.