2-1993

Graphical Evolution Experiments in Artificial Life

Gary R. Greenfield

Graphical Evolution Experiments in Artificial Life

Gary R. Greenfield
Department of Mathematics and Computer Science
University of Richmond
Richmond, Virginia 23173
email: ~~grg5l@aurora.urich.edu~~
ggreenfi@richmond.edu

February, 1993

TR-93-01

1 Introduction

The field of artificial life, or simply "alife" as it has become known to its practitioners, owes its existence and its development to Christopher Langton. As Levy's pop history of alife documents [8], Langton picking up the cold, dormant trail left by Ulam, Von Neumann, and to a lesser extent Conway and the "Game of Life" cult, created an easy to fathom self-reproducing cellular automaton and, to underscore its significance, then proceeded to axiomatize principles for artificial life. Further, under the auspices of the Santa Fe Institute for Nonlinear Studies, Langton organized interdisciplinary conferences which served both to identify alife investigators and to collect and organize their body of work in invaluable, if lengthy, proceedings [6] [7].

To over simplify, the rationale for alife runs as follows: If one wants to study the origins and development of life, there is one and only one example to study — earthbound, carbon based, DNA inspired biological life, known as *wet life.* To find other examples one must look to artificial or synthetic life, usually realized in terms of computer simulations through which one hopes to distill, isolate, and control the basic tenets of growth, metabolism, evolution, behavior, etc. Thus artificial life becomes the antithesis of artificial intelligence — the former trying to identify the simplest and most elementary components necessary for complex processes while the latter tries to mimic, even surpass, complex processes. As one wag has put it, "Why should we expect to be able to build a robot with more functionality than a human when we can't build a robot with more functionality than an earthworm?" Our purpose is not to survey the scope and history of alife, the reader is referred to the Alife Proceedings for that, so we shall begin to narrow our focus by considering the genotype problems alife confronts.

The point of view adopted by many alife researchers is that since genes determine traits, an alife implementation need only specify the traits to be studied and then rely on data structures to maintain and modify those traits. An obvious way to do this is to view the traits as *parameters*, and an arguably sound way to parameterize traits is via (bit) strings. Fortunately, or unfortunately as the case may be, this allows the alife researcher to use the tools developed by researchers in the field of genetic algorithms, a subdiscipline of computer science.

Genetic algorithms automate algorithm development by evolving and evaluating algorithms on the basis of their ability to perform a *specific task.* Again, a survey of genetic algorithms is beyond our scope, but [10] is a comprehensive up to date reference. The point is, that from genetic algorithms

one can usurp a host of techniques for structuring and evolving bit strings that are treated as a genotype of parameters. To furnish an example, we shift our attention to computer graphics.

Two fundamental problems the alife researcher must confront are the problem of understanding what is actually taking place during the alife simulation, and the problem of interpreting the results of the simulation. How does one collect and make sense of data from a simulation that encounters billions of "organisms" per run? Scientific visualization and supercomputers are a must. The most publicized and spectacular alife experiments (for example those of Hillis or Jefferson as recounted by Levy [8]) are vivid proof of this. The trail we want to follow, blazed by Dawkins [1], combines the tasks of genotype representation together with understanding the simulation by making the organism itself a visual entity! The details are of interest. Dawkins posits a genotype with nine genes controlling embryological traits with names like "segment-number" and "segment-distance." The gene values are decimal integers, so the genotype is a decimal string, and the gene values drive a recursive algorithm that generates the visual organism or *biomorph.* It is straightforward to observe how perturbing the genotype string "evolves" the organism.

Dawkins' seminal notion of using a genotype string to drive a rendering algorithm has been exploited on graphics merits alone. Todd, providing technical expertise, and Latham, providing artistic direction, developed their Mutator System for allowing an artist to evolve textured and ray-traced 3D organisms through a user interface that directly manipulates the gene strings [12] [13]. Since there are 24 parameterized genes to control, one thinks of the artist as selectively evolving organisms while exploring or journeying through 24 dimensional parameter space. Another instantiation of this idea, this time for sculpture, is described by McGuire [9].

A departure from fixed length, parameterized genotype strings for creating visual organisms was developed by Sims [11]. The technique, described more fully in the next section, was again implemented in a system whereby users selectively control the evolution of organisms. Thus neither Todd and Latham nor Sims ever attempted a true alife simulation using their visual entities! We seek to remedy this oversight by describing a series of alife experiments with visual organisms designed according to a variant of Sims' genotype representation. Before doing so we should mention the one true alife simulation whose visual aspects influenced us.

Larry Yaeger's alife simulation running on a Silicon Graphics Iris Workstation is called PolyWorld. Our description of PolyWorld is based on notes

taken during an oral presentation and video demonstration given in the Artificial Life Panel Session of SIGGRAPH '92. In PolyWorld the visual organisms roam on a *bounded* two dimensional grid. The organisms "brains" are small neural nets enabling the organisms to control their external visual appearance and to perceive the external world by processing pixmaps. The simulation controls for total energy while striving to explore competition and self-organization. Genes present are for size, strength, maximum speed, mutation rate, number of crossover points in the neural net, lifespan, energy to offspring, and ID (a parameter used to enable mimicry). The neural net can make decisions about whether the organism should eat, fight, mate, move, turn, light (effecting the external appearance of the light sensor panel it emits), or focus (gaze at the appearance of others). To see the organisms evolve to different species adopting distinct and atypical strategies and behaviors for survival is most impressive. Words do not do justice to the video animation sequences.

2 Graphical Alife

In this section we describe our implementation of Sims' mutating expressions [11]. The backbone of our implementation was first developed for a computer art application [3]. The important *differences* to note in our implementation are: (1) Where Sims uses neighborhood image processing functions (*e.g.*, warping, blurring, dissolving, filtering, and noise generating functions) we use point processing functions which are always *normalized* so that inputs and outputs lie in the unit interval and (2) When visualized, our organisms are rendered relative to a `cmax` valued color map as opposed to an absolute *rgb* scale. Unfortunately, the sequence of experiments described in the next section did not progress to the point where environment and organisms were rendered in color.

Let $\mathcal{V}$ be a set of algebraically independent variables, let I be the unit interval, and for $i = 1, \ldots, n$ fix an integer $d_i > 0$, a function $f_i : I^{d_i} \longrightarrow I$, and let $\mathcal{F} = \{f_i\}$.

DEFINITION. An *expression* is defined recursively as a node (a, b, g) where $a, b \in I$, and either $g \in \mathcal{V}$ or $g \in \mathcal{F}$. When $g \in \mathcal{F}$, the arguments of g are again nodes.

We will use $\mathcal{V} = \{c, u, v, t\}$, where c represents a *constant*, u and v parameterize the unit square, and t is a time variable. We only require mappings

f_i defined on I, I^2, and eventually I^3 (*viz.*, $d_i = 1$, 2, and eventually 3). Thus, until further notice $\mathcal{F}$ consists of functions from the unit interval to itself with one argument named

{nsin, ncos, nexp, nlog, nabs, nsqt, nsqr, ncub, nneg}

together with functions from the unit square to the unit interval with two arguments named

{nadd, nmul, nmod, nmin, nmax, npwr, nand, nvee, ncir}.

The precise definitions for these functions are not relevant, but the mnemonics will give clues about their construction.

EXAMPLE. Omitting outermost parentheses for expressions, suppressing commas between arguments, and remembering that **nsin**, **ncos**, and **nsqt** are suitably *normalized* sine, cosine, and square root functions in one variable, a legal expression which up to coefficients calculates the minimum of the sine of u and the cosine of the square root of v is

```
0.05 0.38 nmin(0.54 0.09 nsin(0.42 0.52 u)  0.03 0.69
ncos(0.22 0.40 nsqt(0.04 0.63 v))).
```

DEFINITION. An expression is *evaluated* at (t_0, u_0, v_0) using the recursive evaluation rule $\mathcal{E}$ given by

$$\mathcal{E}(a, b, g) = a * \mathcal{E}(g) + b \pmod 1$$

where if $g \in \mathcal{V}$,

$$\mathcal{E}(g) = \begin{cases} 0 & \text{if } g = c \\ t_0 & \text{if } g = t \\ u_0 & \text{if } g = u \\ v_0 & \text{if } g = v \end{cases}$$

and if $g = f(n_1, \ldots, n_d)$, where $f \in \mathcal{F}$ and $n_1, \ldots, n_d$ are the argument nodes of f,

$$\mathcal{E}(g) = f(\mathcal{E}(n_1), \ldots, \mathcal{E}(n_d)).$$

To summarize, our organisms are computational entities that can also be visualized. We next describe their evolutionary and mating capabilities. A "creation" event provides a pseudorandom expression, used as a starting genotype, that is very primitive, perhaps something as mundane as

```
0.85 0.38 ncir(0.80 0.35 v  nneg(0.42 0.52 u)).
```

This primitive genotype is "evolved" to provide the complex genotype for a *species*. This process is accomplished by successively feeding the genotype to an algorithm that complexifies it by evolving nodes of type $\mathcal{V}$ into those of type $\mathcal{F}$ on a probabilistic basis, and similarly evolving some one argument nodes into two argument nodes. The probablistic genotype evolution algorithm has many decision table parameters and is "intelligent" enough to try and preserve the history of the genotype by promoting evolution near the leaves rather than near the root of the expression. A typical species genotype that might arise follows.

```
0.87 0.56 nmul(0.80 0.63 nsin(0.85 0.95 nsqt(0.80
0.62 u))  0.87 0.75 nexp(0.80 0.17 nmin(0.80 0.40 v
0.80 0.47 u)))
```

All organisms of the species will use this genotype, meaning their expressions will have this identical tree structure.

The initial population of the species consists of `NORGANISMS`, a third of which have *all* their nodes randomly selected from the $\mathcal{V}$ and $\mathcal{F}$ sets while still remaining faithful to the the genotype; a third of which have a randomly selected proportion of their nodes determined in this fashion; and a third third of which have all arguments faithful to the species genotype but with the a and b coefficients at each node severely perturbed. Our alife simulation follows the development of this population, or *colony*, through a prescribed number of time cycles, say `NCYCLES`.

The organisms can reproduce sexually or asexually. The sexual reproduction, or mating, algorithm for two organisms creates an offspring organism faithful to the species genotype node by node with reference to a parameter that biases node selection in favor of a dominant organism. Asexual reproduction merely perturbs the node's a and b coefficients as described above. All organisms are subject to mutation which can alter the arguments of one or more nodes, and all organisms undergo "genetic drift" which is implemented by a very mild perturbation of each node's a and b coefficients. There are many more aspects of the simulation to discuss, but they will be dealt with as they arise in the context of the sequence of experiments that was actually performed.

3 The Experiments

Experiment #1.

In general, our organisms are *camouflage* organisms whose survival depends on their ability to blend in with their surroundings. For this first experiment, we challenged them to breed to match a simulated background. Since the organisms perform computations on the unit square via the u, v parameters, we assigned random values $r_{i,j}$ to the following nine "sites" of the unit square:

$$\{(u_i, v_j) : u_i = i/3, v_j = j/3, 0 \leq i, j \leq 2\}.$$

We used the sum of the squares of the errors to these target values as the fitness criterion:

$$\mathcal{C}(g) = \sum_{i,j} (\mathcal{E}(g)|_{(u_i,v_j)} - r_{i,j})^2.$$

The population was held steady at 100 organisms. After each time cycle, the best 20 organisms were retained for breeding. The most fit organism was mated with the other 19, and the second most fit organism was mated with the other 18. The population was then restored to 100 organisms by allowing all the breeding stock to contribute *asexually* produced offspring. Note that the time variable t was not used in our early experiments.

We have mentioned that every organism is subject to genetic drift and mutation. When an organism is created a random number is compared to the mutation rate, `mrate`, to decide whether or not a randomly selected node should be altered. The mutation rate begins at 0.20 but is adjusted during the course of the simulation so that it may drop as low as 0.02 or soar as high as 0.40. At the completion of each time cycle, the rate changes by 0.01 according as the most fit individual present — the one with the greatest influence on the next generation — is better at camouflage than the most fit individual of the previous generation. Our typical run was set for 100 time cycles.

We observed that by the end of a run the most fit organism achieved a fitness of about 0.05 and the mutation rate hovered near 0.10. Surprisingly, almost always, superior organisms were computing using only constants and one of the u and v variables! Examining sample sets of target values suggested an explanation for what was happening. The organisms had taken advantage of the fact that many sites had similar target values. With limited computational resources an organism was rewarded for focusing its attention to the variable most responsible for target variation, and the sum of squares fitness favored a genetic attempt at one variable approximation to cover this spread. We viewed the breeding stock at the end of each run

and observed that while node structures were nearly identical with respect to arguments, indicating that the breeding stock was inbred, the modest genetic drift (a *maximum* of 0.05 per coefficient) was severely effecting the fitness. Typically the least fit member of the breeding stock had fitness of about 0.25. We expected tighter clustering of the fitness values.

Our simulation does not allow node coefficients to "wrap," so an a coefficient drifting toward 1.00 cannot wrap to 0.00 and an a coefficient drifting towards 0.00 cannot wrap to 1.00. Node coefficients, especially dominant coefficients close to the root of the expression, had an annoying tendency to drift to extremes — 0.00 for a and 1.00 for b, or conversely — indicating that successful strategies often resulted from ignoring variables or ignoring constants. Again the distribution of target values was critical.

Experiment #2.

To promote greater diversity in the population and to try and counteract how quickly the simulation was reaching steady state, we increased the complexity of the genotype by feeding it through the genotype evolution algorithm 20 times as opposed to 10. We increased the initial mutation rate from 0.20 to 0.33, we eliminated the dominance factor in sexual mating, and we permitted more dramatic mutations by altering *two* node arguments whenever mutations were called for. But the key change was to view organism camouflage sites not as fixed points but as areas which would respond to point stimuli. Thus an organism was envisioned as trying to camouflage itself by matching as best it could random values assigned to each of the nine $1/3 \times 1/3$ uniformly sized patches or sites of which it was made up. To evaluate fitness, a random point was selected in each patch, and its computational value was compared to the target value for that patch. The random points clustered around the center of the site so the organism had a better chance to make graceful transitions from site to site.

We indeed found that organisms had a much more difficult time with the camouflage task. It was rare to see the mutation rate drop below 0.25 during the course of a run. The variation in target values coupled with being kept off balance with respect to site queries meant that in every time cycle the majority of the colony was poorly camouflaged, while a few lucky organisms were well camouflaged. We suspected that the latter was due to being queried at the "right" set of site points.

Experiment #3.

At this point, we implemented the truly unique feature supported by our

alife design, an environment for the organisms to live in that could also mutate and evolve! The environment was also created using the genotype evolution algorithm. This meant our simulation bore some semblance to a "Gaia model" or perhaps a colony parasitic on an individual of another species. Hypothesizing that the environment would be perceived as relatively stable in comparison with our short lived camouflage species, we only allowed the environment to alter itself by genetic drift using the miniscule drift limit setting of 0.01. The environment was treated as a 10×10 organism. (Recall that camouflage organisms are treated as 1×1 organisms.) We reduced the colony size to 25 and set the mutation rate back to 0.20. The organisms now had 25 patches or sites. Randomly selected points within *perimeter* sites were used to calculate camouflage values (after all, why should one be able to see the interiors of these two dimensional organisms?), and these were now compared to the environment organism's values computed at the same points. Obviously, scaling and translation were necessary to "align" organism and environment. The x and y coordinates used to determine the organism's location were at the organism's discretion because interior sites were reserved for their computation. All reproduction was by "cloning" so one would expect a newly cloned organism to be located close to its parent because its position calculation apparatus would be inherited. After each time cycle the best five camouflaged organisms were cloned twice, and the worst five were not cloned at all. Henceforth in our experiments the mutation rate fluctuated at each time cycle according to the *average* fitness of the colony.

This first experiment using an environment organism was primarily a "proof of concept." The colony camouflage rating and mutation rate quickly stabilized, and examining the top ten individuals after each run revealed that organisms clustered at the "boundary" of the environment or along the diagonal.

Experiment #4.

Yet another "proof of concept" experiment. Now we allowed our organisms to have a *lifespan.* We increased the population size to fifty, and permitted an organism to live to a maximum age of ten. This was not at easy as it sounds. At the end of each time cycle, juveniles, those under age two, were "protected," the least fit *adults* perished, and the breeding population – the most fit adults aged two through eight – were cloned to bring the population back to full strength. The initial population had random age assignments.

We were pleased to observe that the average age of the colony quickly reached steady state. The organisms still clustered, staking out positions on the boundaries, on horizontal or vertical strips, and on the diagonal.

Experiment #5.

This experiment relaxed the camouflage calculation by querying only the four corner sites together with the middle sites at each edge. In future experiments we planned to use the remaining perimeter sites for communication with the external world. But the primary purpose of this experiment was to initiate some movement on the part of the colony by using a random point within the designated location sites to (re)calculate the organism's x and y coordinates at each time cycle. We increased the size of the colony to 70 and the number of time cycles to 75, but the colony remained very static and clustered at the diagonal. The fact that organisms clumpeded at all was viewed positively — similar to discovering a well camouflaged patch of mushrooms or nest of insects.

Experiment #6.

Having assigned random ages to the initial population, we now also randomly assigned initial locations to the population, and we forced the organisms to "explore" their environment by using their location sites to compute x and y *displacements.* For the time being we left the decision of whether displacements should be positive or negative to chance. The distance per time cycle an organism could travel could not exceed its "length." Newly cloned organisms took their place at the side of their parent. Without a graphics display it was difficult to understand the simulation, even with as few as 20 organisms and 10 time cycles (one lifespan).

Experiment #7.

We hooked our alife simulation into the SRGP graphics package [2] allowing us to view the position of each organism — actually the lower left hand corner of each organism superimposed on a display scaled so that the environment was 300 x 300 pixels — after each time cycle. Moreover, mutant organisms were specially tagged. Since an organism might traverse its environment in a single lifespan, we made the environment toroidal allowing its offspring to duplicate its migratory path. A computation site was designated for the organism to determine a compass direction to use in conjunction with its displacements. Here we restricted the directions to the four cardinal points of the compass. Organisms were allowed to incorporate the time variable into their genotype, but the time variable was only for

camouflage organisms not for the "timeless" environment organism. The organism's age served as the computational input for time. Also, organisms were permitted to use "logic" via simple-if, `nsif`, or if-then-else, `nsel`, nodes. The conditional component of the expression evaluated to `TRUE` when the computational value was greater than one-half, and to `FALSE` otherwise (a kind of fuzzy logic). For simple-if, `FALSE` meant the organism computation yielded 0.0, but in all other cases the natural computational value was returned.

As expected the display revealed that there were zones within the environment where the organisms could flourish. But even with movement slowed so organisms could move at most half a body length per time cycle motion patterns were confusing.

Experiment #8.

We slowed the movement so that even under maximum speed in a consistent direction it would take an organism 50 cycles to traverse its environment. We also allowed the organisms to travel diagonally, thus organisms could now choose from eight compass directions. We increased the lifespan of an organism to 20 cycles, and we re-populated the colony at the end of each cycle by mating randomly selected pairs of the breeding population. Offspring location was set so that a newborn would accompany one of its parents. In this model every member of the breeding population disperses its genotype, and some collect this reproductive material for sexual union and birth. This is reminiscent of fertilization in tide pools or coral reefs. Graphics were enhanced by using distinct markers for juveniles, mutants, and adults.

The simulation was allowed to continue for up to 900 cycles during some runs. There was considerable ebb and flow observed in camouflage average and in the movement of our colony of 50 organisms.

Experiment #9.

We wanted to make the colony more assertive than its environment; aggressive learners and evolvers if you will. Thus we designed functions supporting logical computation, denoted `bsif`, `bleq`, and `bgeq` for *boolean* simple-if, less-than-or-equal, and greater-than-or-equal. (Warning: The `nand` function mentioned earlier is calculated on the basis of the bit pattern of its two arguments). These new boolean two argument functions returned fuzzy truth values: for simple-if, a fuzzy `FALSE` otherwise the computational result; and a fuzzy `TRUE` or `FALSE` as necessary for the inequalities. The en-

vironment organism was prevented from using the time parameter or these fuzzy boolean nodes, but colony organisms could enjoy both. All organisms could use the boolean if-then-else, now renamed `bsel`. We also forced the organisms to rely more heavily on their computational machinery by enforcing a lower bound of 0.80 on the a coefficient at each node. Finally, we enlarged the environment to a 100×100 world for our 1×1 organisms. Conceivably, the environment could still be crossed in fifty time cycles.

We have avoided cluttering experiment descriptions with computer printouts. But appended is a sample graphics display from the end of a 200 time cycles run for a colony of 50 organisms. It is accompanied by printout that gives the static genotype for the environment, the initial template genotype for the colony, and after every tenth time cycle the average age of the colony (as a percentage of maximum lifespan), the average camouflage value, and the current mutation rate. The printout summarizes this data for every individual in the colony at the end of the run, and then provides the genotype of every *fifth* organism in the colony as ranked by camouflage value.

4 Open Questions, Future Work

Our alife simulation experiments have raised almost as many questions as they have answered. We should first remark that though the genotype of our organisms bears a resemblance to that of Koza [4] [5], the inspiration clearly comes from Sims [11]. For Koza, the tree structures parse to pseudocode and there is no concept of a species genotype, hence *all* tree structures are templates for the population. More to the point, Koza's organisms can evolve to a "perfect form" by, say, successfully running a maze or reproducing a logic table. We are unable to define perfect organisms. Our use of a template genotype seems natural but raises a vexing question.

Question. How does a species complexify its genotype? Specifically, how could our simulation increase the size and complexity of the genotype "on the fly" and still avail itself of the node-by-node method for sexual reproduction that seems most natural?

The reason it is important to try and provide answers to these questions is because only by complexifying the genotype will a species be able to build and pass on a "genetic memory." It may be seen as a shortcoming of our model that our organisms do not "learn" during their lifespan. For now we view the organisms as too primitive to *act* on external stimuli and record

information about such actions. We have considered ways to incorporate such notions, but to speculate further on them at this time seems wholly premature.

Clearly, the graphics for our simulation need improvement. First, the environment should be visualized so we can "see" what the organisms must contend with. Second, we would like to be able to see if the organisms are exploiting the fact that exterior sites are for camouflage and interior sites are discretionary. Organism and environment visualization would immediately give clues about the wisdom of using an environment to organism length ratio of 10 versus 100. Considerable compute power seems called for in the future. Perhaps the simulation should even be performed on a supercomputer. There is also a technical visualization question to consider.

Question. Our organisms are now permitted to use fuzzy logic. Can our visualization algorithm cope with this?

We need to devote more effort to understanding what our organisms are computing. Certainly we need tools to visualize their genotypes in tree form. This might make their computation strategies apparent or reveal how much or little computational control they exhibit. This would also provide indications about how the parameters in $\mathcal{V}$ are being used, and suggest what a reasonable genotype size should be for supporting 25 computational sites.

Future work should include a series of comparison experiments to explore how useful fuzzy logic functions are, and how utilitarian the core set of functions $\mathcal{F}$ is. It would make sense to try several different species genotypes on the same environment in combination with several $\mathcal{F}$ designs. We should even experiment with other camouflage measures!

Question. For evolution or alife purposes, how should one best measure how well n fitness values $e_1, \ldots, e_n$ estimate or approximate n target values $t_1, \ldots, t_n$?

Several very hard alife problems are still to be addressed. How large should the population be? How long should the simulation run? How do we let decisions about reproduction emerge? Several standard models base these types of decisions on environmental resource measures and individual fitness measures referred to as individual health or energy. We have not considered trying to design food consumption and food supply mechanisms because we believe camouflage is a different paradigm, one supported by a

resource laden environment. In fact, it might make more sense for us to design an ecosystem, perhaps consisting of a large, slowly dispersing camouflage species and a small, fast moving predator species (think chitons and starfish).

Such a scenario leads us to conclude by re-emphasizing the strengths of our model to help justify why we think it has future alife potential: Environment and life forms are governed by a consistent "biology" – mutating expressions – that can adapt, mutate, evolve and be set to any task that we are willing to formulate as purely computational.

References

[1] Richard Dawkins, The evolution of evolvability, 201–220, *Artificial Life*, Chris Langton (ed.), Addison-Wesley, Reading, MA 1989.

[2] James Foley, Andries van Dam, Steven Feiner & John Hughes, *Computer Graphics — Principles and Practice*, Addison-Wesley, Reading, MA 1990.

[3] Gary Greenfield, Graphical evolution and computer art, preprint.

[4] John Koza, Graphical evolution and co-evolution of computer programs, 610–617, *Artificial Life II*, Chris Langton (ed), Addison-Wesley, Reading MA 1992.

[5] John Koza, A hierarchical approach to learning the boolean multiplexer function, 171–192, Gregory Rawlins (ed), *Foundations of Genetic Algorithms*, Morgan Kaufmann, San Mateo, CA 1991.

[6] Chris Langton (ed), *Artificial Life, The Proceedings of an Interdisciplinary Workshop on the Synthesis and Simulation of Living Systems held September, 1987 in Los Alamos, New Mexico*, Santa Fe Institute Studies in the Sciences of Complexity, Vol. VI, Addison-Wesley, Reading MA 1989.

[7] Chris Langton, et al (ed), *Artificial Life II, The Proceedings of the Workshop on Artificial Life held February, 1990 in Santa Fe, New Mexico*, Santa Fe Institute Studies in the Sciences of Complexity, Vol. X, Addison-Wesley, Reading MA 1992.

[8] Steven Levy, *Artificial Life: The Quest for a New Creation*, Pantheon Books, New York, NY 1992.

[9] Frank McGuire, The origins of sculpture: evolutionary 3D design, *IEEE Computer Graphics and Applications*, (January 1993) 9–11.

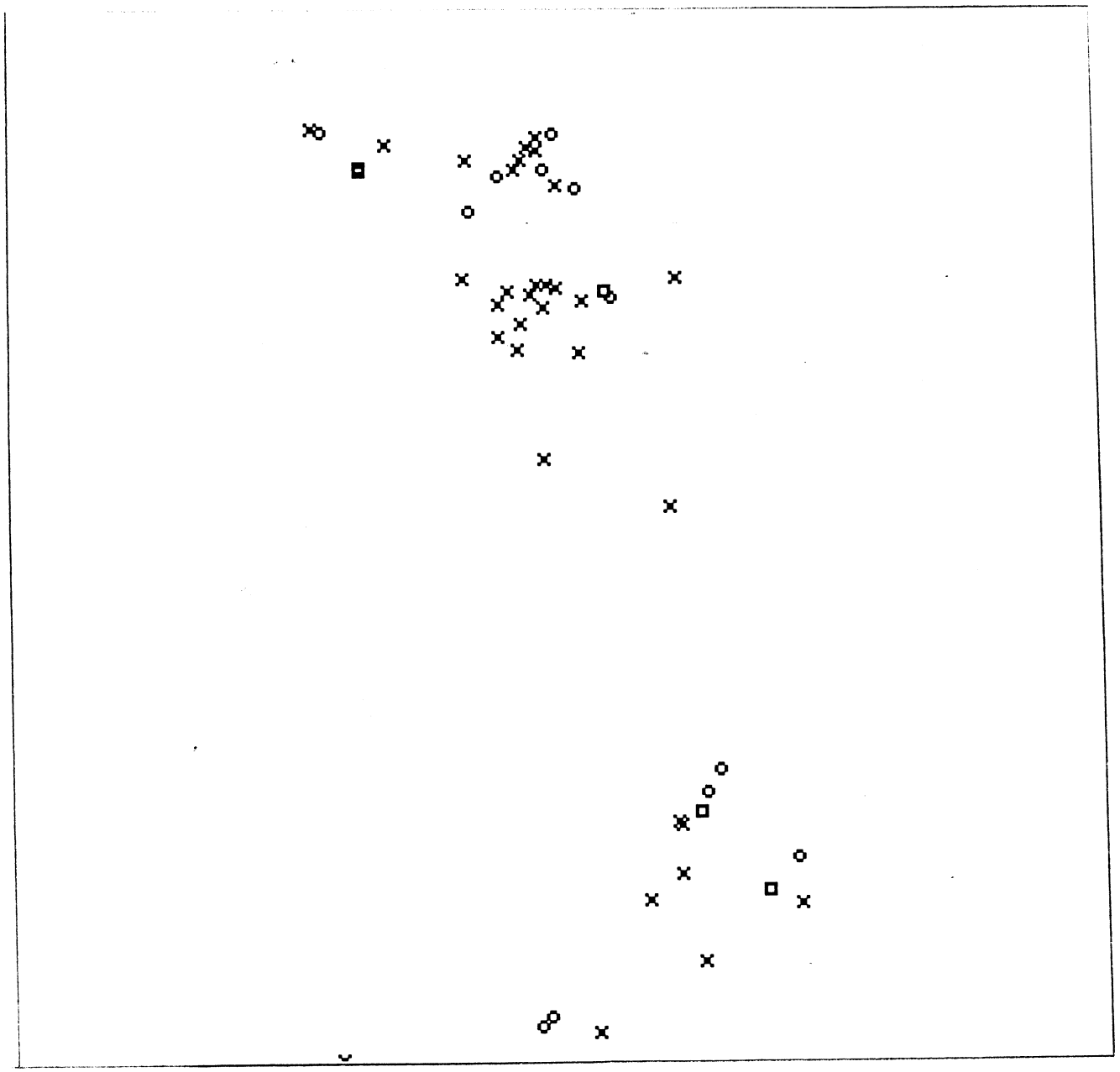
[10] Gregory J. E. Rawlins (ed), *Foundations of Genetic Algorithms*, Morgan Kaufmann, San Mateo, CA 1991.

[11] Karl Sims, Artificial evolution for computer graphics, *Computer Graphics*, **25** (1991) 319–328.

[12] Steven Todd & William Latham, *Evolutionary Art and Computers*, Academic Press, San Diego, CA 1992.

[13] Steven Todd & William Latham, Mutator, a subjective interface for evolution of computer sculpture, IBM UKSC Report 248, 1991.

x - adult
o - mutant
□ - juvenile



A1. Organism Distribution

```
environs
   > 0.80 0.02 ncir(0.80 0.97 nsqr(0.80 0.77 u)  0.80 0.96
     nvee(0.80 0.25 npwr(0.80 0.29 u  0.80 0.71 u)  0.80 0.63
     nneg(0.80 0.84 v)))

genotype
   > 0.80 0.84 bsel(0.80 0.06 nmin(0.80 0.70 nexp(0.80
     0.35 nadd(0.80 0.62 t  0.93 0.61 nneg(0.80 0.62 t)))  0.89
     0.43 nsqt(0.80 0.12 u))  0.80 0.01 bsel(0.80 0.67 nvee(
     0.80 0.99 v  0.80 0.72 v)  0.80 0.42 ncir(0.80 0.91 bsel(
     0.80 0.79 u  0.83 0.48 nsqr(0.80 0.30 c)  0.80 0.37 c)
     0.80 0.54 bsel(0.80 0.18 t  0.80 0.67 t  0.80 0.33 t))
     0.80 0.50 t)  0.80 0.78 nand(0.80 0.90 v  0.80 0.34 v))

cycle =   0    tavg = 0.5210    cavg = 1.0849    mrate = 0.19
cycle =  10    tavg = 0.3400    cavg = 0.5277    mrate = 0.15
cycle =  20    tavg = 0.3880    cavg = 0.4366    mrate = 0.11
cycle =  30    tavg = 0.3710    cavg = 0.5516    mrate = 0.13
cycle =  40    tavg = 0.3970    cavg = 0.4708    mrate = 0.13
cycle =  50    tavg = 0.3790    cavg = 0.4728    mrate = 0.15
cycle =  60    tavg = 0.3620    cavg = 0.3439    mrate = 0.09
cycle =  70    tavg = 0.4040    cavg = 0.5391    mrate = 0.13
cycle =  80    tavg = 0.4000    cavg = 0.6935    mrate = 0.15
cycle =  90    tavg = 0.3810    cavg = 0.6123    mrate = 0.17
cycle = 100    tavg = 0.4130    cavg = 0.4337    mrate = 0.13
cycle = 110    tavg = 0.3960    cavg = 0.4039    mrate = 0.13
cycle = 120    tavg = 0.3760    cavg = 0.3974    mrate = 0.13
cycle = 130    tavg = 0.3730    cavg = 0.4217    mrate = 0.13
cycle = 140    tavg = 0.3830    cavg = 0.2961    mrate = 0.13
cycle = 150    tavg = 0.3610    cavg = 0.5042    mrate = 0.15
cycle = 160    tavg = 0.4040    cavg = 0.5060    mrate = 0.17
cycle = 170    tavg = 0.3160    cavg = 0.7278    mrate = 0.21
cycle = 180    tavg = 0.4040    cavg = 0.7435    mrate = 0.21
cycle = 190    tavg = 0.3970    cavg = 0.6708    mrate = 0.21
```

A2. Environment and Species Genotype; Simulation Averages

i = 0	xval = 0.5715	yval = 0.7902	tval = 0.0500	cval = 0.0008
i = 1	xval = 0.6439	yval = 0.2453	tval = 0.4000	cval = 0.0337
i = 2	xval = 0.3419	yval = 0.9476	tval = 0.6000	cval = 0.0368
i = 3	xval = 0.6110	yval = 0.1683	tval = 0.4000	cval = 0.0474
i = 4	xval = 0.5423	yval = 0.7327	tval = 0.6500	cval = 0.0545
i = 5	xval = 0.5589	yval = 0.0311	tval = 0.8000	cval = 0.1458
i = 6	xval = 0.7680	yval = 0.1662	tval = 0.5500	cval = 0.1468
i = 7	xval = 0.4633	yval = 0.9124	tval = 0.3500	cval = 0.1554
i = 8	xval = 0.4998	yval = 0.8031	tval = 0.6000	cval = 0.1708
i = 9	xval = 0.4207	yval = 0.8095	tval = 0.5000	cval = 0.1822
i = 10	xval = 0.5181	yval = 0.9044	tval = 0.5500	cval = 0.1861
i = 11	xval = 0.6340	yval = 0.5708	tval = 0.8000	cval = 0.2092
i = 12	xval = 0.5130	yval = 0.0437	tval = 0.0500	cval = 0.2493
i = 13	xval = 0.4897	yval = 0.9464	tval = 0.6500	cval = 0.2520
i = 14	xval = 0.4974	yval = 0.9558	tval = 0.2000	cval = 0.2592
i = 15	xval = 0.6422	yval = 0.8091	tval = 0.8500	cval = 0.3019
i = 16	xval = 0.2930	yval = 0.0055	tval = 0.9500	cval = 0.3080
i = 17	xval = 0.5431	yval = 0.8971	tval = 0.2000	cval = 0.3606
i = 18	xval = 0.6426	yval = 0.2494	tval = 0.3000	cval = 0.3714
i = 19	xval = 0.4688	yval = 0.7943	tval = 0.5000	cval = 0.3745
i = 20	xval = 0.5181	yval = 0.7999	tval = 0.2000	cval = 0.3843
i = 21	xval = 0.4984	yval = 0.9433	tval = 0.3000	cval = 0.3911
i = 22	xval = 0.2634	yval = 0.9645	tval = 0.6500	cval = 0.3945
i = 23	xval = 0.2800	yval = 0.9577	tval = 0.0500	cval = 0.4037
i = 24	xval = 0.7389	yval = 0.1758	tval = 0.1000	cval = 0.4605
i = 25	xval = 0.6674	yval = 0.1033	tval = 0.3000	cval = 0.4623
i = 26	xval = 0.5027	yval = 0.0343	tval = 0.3500	cval = 0.5374
i = 27	xval = 0.4917	yval = 0.7910	tval = 0.1500	cval = 0.5463
i = 28	xval = 0.5045	yval = 0.6221	tval = 0.8000	cval = 0.5825
i = 29	xval = 0.4258	yval = 0.9321	tval = 0.2000	cval = 0.5878
i = 30	xval = 0.4307	yval = 0.8766	tval = 0.3000	cval = 0.6574
i = 31	xval = 0.4734	yval = 0.9222	tval = 0.3500	cval = 0.6922
i = 32	xval = 0.6676	yval = 0.2558	tval = 0.1000	cval = 0.7046
i = 33	xval = 0.6879	yval = 0.2973	tval = 0.4500	cval = 0.7260
i = 34	xval = 0.5069	yval = 0.8025	tval = 0.5500	cval = 0.7268
i = 35	xval = 0.5035	yval = 0.7776	tval = 0.2500	cval = 0.7799
i = 36	xval = 0.4806	yval = 0.9308	tval = 0.6000	cval = 0.7819
i = 37	xval = 0.4785	yval = 0.7365	tval = 0.1500	cval = 0.7892
i = 38	xval = 0.6465	yval = 0.1945	tval = 0.4000	cval = 0.8894
i = 39	xval = 0.6749	yval = 0.2759	tval = 0.7000	cval = 0.9375
i = 40	xval = 0.5460	yval = 0.7860	tval = 0.2500	cval = 1.1016
i = 41	xval = 0.4586	yval = 0.7499	tval = 0.7500	cval = 1.1476
i = 42	xval = 0.4589	yval = 0.7805	tval = 0.3500	cval = 1.1509
i = 43	xval = 0.5069	yval = 0.9183	tval = 0.1500	cval = 1.1922
i = 44	xval = 0.5767	yval = 0.7839	tval = 0.1500	cval = 1.2377
i = 45	xval = 0.3183	yval = 0.9197	tval = 0.0500	cval = 1.4635
i = 46	xval = 0.5169	yval = 0.9542	tval = 0.1000	cval = 1.5476
i = 47	xval = 0.3183	yval = 0.9233	tval = 0.1000	cval = 1.6117
i = 48	xval = 0.4814	yval = 0.7610	tval = 0.3000	cval = 1.6427
i = 49	xval = 0.7692	yval = 0.2078	tval = 0.4500	cval = 1.7804

A3. Individual Organism Data

```
> 0.88 0.89 bsel(0.80 0.91 ncir(0.80 0.98 nexp(0.95
  0.09 nmax(0.82 0.91 t  0.82 0.38 nneg(0.87 0.65 v)))  0.86
  0.67 nneg(0.80 0.33 u))  0.80 0.08 bsel(0.80 0.20 nvee(
  0.93 0.97 t  0.90 0.62 v)  0.87 0.59 ncir(0.90 0.15 bsel(
  0.99 0.86 u  0.80 0.48 nsqr(0.84 0.63 v)  0.84 0.22 v)
  0.89 0.38 bsel(0.80 0.82 t  0.82 0.36 t  0.80 0.07 u))
  0.83 0.76 u)  0.95 0.34 nmod(0.80 0.18 t  0.91 0.82 u))

> 0.84 0.96 bsel(0.90 0.12 ncir(0.80 0.81 nexp(0.82
  0.38 nmax(0.87 0.72 u  0.86 0.72 nneg(0.86 0.97 t)))  0.80
  0.55 nneg(0.90 0.51 u))  0.80 0.05 bsel(0.92 0.67 nvee(
  0.80 0.82 v  0.84 0.79 v)  0.80 0.60 ncir(0.88 0.77 bsel(
  0.88 0.56 t  0.92 0.53 nsqr(0.80 0.49 u)  0.80 0.23 c)
  0.80 0.80 bsel(0.91 0.99 t  0.80 0.45 c  0.80 0.37 u))
  0.80 0.19 u)  0.83 0.22 nand(0.81 0.80 v  0.80 0.99 u))

> 0.81 0.80 bsel(0.86 0.22 ncir(0.83 0.60 nexp(0.87
  0.19 nmax(0.85 0.87 u  0.85 0.72 nsqr(0.92 0.66 u)))  0.80
  0.48 nsin(0.87 0.89 u))  0.80 0.16 bsel(0.82 0.72 nvee(
  0.85 0.95 v  0.85 0.43 v)  0.94 0.41 ncir(0.96 0.88 bsel(
  0.88 0.65 u  0.80 0.23 nsqr(0.80 0.63 u)  0.80 0.30 v)
  0.80 0.04 bsel(0.80 0.86 t  0.81 0.78 t  0.80 0.74 v))
  0.80 0.31 u)  0.89 0.10 nmul(0.91 0.90 v  0.87 0.45 v))

> 0.95 0.14 bsel(0.88 0.25 nmin(0.97 0.86 nexp(0.80
  0.18 nmax(0.86 0.58 t  0.86 0.40 nneg(0.97 0.39 t)))  0.87
  0.54 nneg(0.89 0.38 u))  0.83 0.06 bsel(0.92 0.43 nvee(
  0.88 0.70 t  0.96 0.90 v)  0.90 0.53 ncir(0.80 0.17 bsel(
  0.89 0.69 u  0.90 0.51 nsqr(0.80 0.31 u)  0.80 0.47 v)
  0.80 0.29 bsel(0.86 0.92 t  0.89 0.27 u  0.89 0.38 t))
  0.85 0.37 v)  0.84 0.43 nmod(0.83 0.29 t  0.89 0.99 u))

> 0.84 0.91 bsel(0.98 0.95 nand(0.80 0.96 nexp(0.85
  0.15 nmax(0.80 0.92 t  0.86 0.40 nneg(0.80 0.42 t)))  0.87
  0.46 nneg(0.80 0.28 u))  0.82 0.09 bsel(0.97 0.21 nvee(
  0.89 0.90 t  0.80 0.97 v)  0.84 0.50 ncir(0.82 0.69 bsel(
  0.80 0.61 u  0.88 0.10 nsqr(0.80 0.69 v)  0.88 0.26 v)
  0.97 0.70 bsel(0.98 0.24 t  0.80 0.31 t  0.88 0.28 u))
  0.82 0.47 u)  0.80 0.24 nmul(0.80 0.89 t  0.86 0.75 u))

> 0.83 0.97 bsel(0.80 0.10 ncir(0.82 0.56 nexp(0.80
  0.26 nmax(0.90 0.48 u  0.81 0.23 nneg(0.80 0.33 t)))  0.80
  0.55 nneg(0.80 0.26 u))  0.86 0.17 bsel(0.83 0.77 bleq(
  0.87 0.71 v  0.94 0.54 c)  0.96 0.46 ncir(0.80 0.04 bsel(
  0.80 0.58 u  0.83 0.24 ncub(0.83 0.32 u)  0.90 0.20 v)
  0.80 0.36 bsel(0.80 0.02 t  0.80 0.53 t  0.84 0.79 v))
  0.85 0.66 u)  0.80 0.40 nadd(0.80 0.06 v  0.80 0.76 u))

> 0.88 0.87 bsel(0.93 0.05 ncir(0.80 0.53 nexp(0.80
  0.58 nmax(0.90 0.46 u  0.93 0.40 ncub(0.99 0.81 u)))  0.92
  0.61 nneg(0.80 0.13 u))  0.80 0.11 bsel(0.80 0.36 nvee(
  0.98 0.52 t  0.96 0.90 v)  0.80 0.46 ncir(0.80 0.79 bsel(
  0.80 0.77 u  0.80 0.38 nsqr(0.80 0.50 t)  0.86 0.08 v)
  0.80 0.25 bsel(0.80 0.73 t  0.80 0.82 t  0.90 0.81 v))
  0.88 0.55 t)  0.80 0.19 nand(0.88 0.02 v  0.80 0.18 v))
```

A4. Selected Individual Genotypes

```
> 0.90 0.76 bsel(0.90 0.35 ncir(0.80 0.59 nexp(0.85
  0.19 nmax(0.92 0.38 t  0.80 0.49 nneg(0.82 0.42 t)))  0.90
  0.64 nneg(0.82 0.88 u))  0.89 0.09 bsel(0.80 0.70 bleq(
  0.89 0.04 t  0.80 0.32 v)  0.80 0.65 ncir(0.87 0.97 bsel(
  0.80 0.66 u  0.89 0.31 nsqr(0.80 0.66 v)  0.87 0.05 v)
  0.83 0.81 bsel(0.80 0.16 t  0.86 0.29 u  0.80 0.22 u))
  0.96 0.67 u)  0.84 0.35 nand(0.96 0.09 t  0.80 0.81 u))

> 0.89 0.99 bsel(0.86 0.12 ncir(0.96 0.62 nexp(0.80
  0.33 nmax(0.92 0.52 t  0.80 0.98 nneg(0.80 0.05 t)))  0.80
  0.39 nneg(0.96 0.88 u))  0.96 0.11 bsel(0.80 0.80 nvee(
  0.84 0.61 v  0.80 0.61 v)  0.90 0.16 ncir(0.80 0.81 bsel(
  0.83 0.66 t  0.87 0.24 nlog(0.95 0.65 v)  0.91 0.25 c)
  0.83 0.93 bsel(0.81 0.83 t  0.98 0.25 t  0.81 0.42 t))
  0.80 0.55 u)  0.95 0.45 nand(0.84 0.10 v  0.93 0.93 u))

> 0.85 0.04 bsel(0.80 0.06 nmin(0.99 0.74 nexp(0.94
  0.44 nmax(0.80 0.17 u  0.80 0.89 nsqr(0.89 0.33 t)))  0.90
  0.55 nneg(0.90 0.93 u))  0.80 0.02 bsel(0.80 0.78 nvee(
  0.94 0.94 t  0.98 0.82 v)  0.90 0.39 ncir(0.99 0.86 bsel(
  0.94 0.84 t  0.80 0.39 nsqr(0.89 0.71 u)  0.88 0.31 c)
  0.95 0.56 bsel(0.80 0.93 t  0.80 0.82 t  0.80 0.45 t))
  0.90 0.81 u)  0.80 0.50 nand(0.84 0.86 t  0.85 0.74 u))

mutations :    173
births    :    995
```

A5. Genotypes continued