Environment as a Spatial Constraint on the Growth of Structural Form

T. Kowaliw
Dept. Computer Science &
Software Engineering
Concordia University
1455 de Maisonneuve Blvd.
Ouest, Montréal, QC, Canada
taras@kowaliw.net

P. Grogono
Dept. Computer Science &
Software Engineering
Concordia University
1455 de Maisonneuve Blvd.
Ouest, Montréal, QC, Canada
grogono@cs.concordia.ca

N. Kharma
Dept. Computer & Electrical
Engineering
Concordia University
1455 de Maisonneuve Blvd.
Ouest, Montréal, QC, Canada
kharma@ece.concordia.ca

ABSTRACT

We explore the use of the developmental environment as a spatial constraint on a model of Artificial Embryogeny, applied to the growth of structural forms. A *Deva* model is used to translate genotype to phenotype, allowing a Genetic Algorithm to evolve Plane Trusses. Genomes are expressed in one of several developmental environments, and selected using a fitness function favouring stability, height, and distribution of pressure. Positive results are found in nearly all cases, demonstrating that environment can be used as an effective spatial constraint on development. Further experiments take genomes evolved in some environment and transplant them into different environments, or re-grow them at different phenotypic sizes; It is shown that while some genomes are highly specialized for the particular environment in which they evolved, others may be re-used in a different context without significant re-design, retaining the majority of their original utility. This strengthens the notion that growth via Artificial Embryogeny can be resistant to perturbations in environment, and that good designs may be re-used in a variety of contexts.

Categories and Subject Descriptors

I.2.8 [**Computing Methodologies**]: Artificial Intelligence—*Problem Solving, Control Methods, and Search*; F.1.1 [**Theory of Computation**]: Computation by Abstract Devices—*Models of Computation*

General Terms

Algorithms, Design, Experimentation

Keywords

Artificial Embryogeny, Developmental Algorithms, Evolutionary Computation, Structure, Environment, Computational Development, Truss, Topological Optimization

1. INTRODUCTION

In this paper, we explore the use of environment as a means of providing additional control to the process of Artificial Embryogeny (AE), and the efficacy of those agents re-developed in foreign environments.

We introduce the *Deva 1.N* model, a slight modification of our earlier Deva 1 model. Deva 1.N is a Cellular Automaton-like model where cells develop in a discrete space and under a discrete time. Deva 1.N is a general means of mapping between genotype and phenotype, where phenotype is realized in a dynamical system guided by the genome. In order to differentiate between the relative merits of our techniques, we use an independent and external means of evaluation; We interpret our organisms as Plane Trusses, evaluated on their ability to form a stable structure and support external forces. Plane Trusses are common models in structural design — successful truss design is a challenging task, relevant to engineering today.

The novel portion of our current research concerns the addition of spatial constraints to the growth process, this through the use of a developmental space. Our phenotypes are expressed in a variety of developmental spaces, controlling their growth through their geometric shape. The evolution of structural form under spatial constraint is desireable for two reasons: Firstly, it is probably much easier to specify spatial constraints on a lattice than it is through the usual means of controlling evolution, a fitness function; Secondly, the imposition of spatial constraints may help to reduce the necessary search space, as extraneous growth in undesirable directions is limited.

Our initial experiments with the various environments were successful, as evolution was able to find stable, load-bearing trusses in nearly all environments specified.

Following this is a set of experiments devoted to studying the re-growth of genomes evolved in some environment, transplanted into another. As a metaphor for re-growth, imagine the following situation: A scientist finds a recently impregnated Bonobo monkey, whose womb contains a single fertilized cell, ready to begin growing into a new Bonobo child. The scientist removes this zygote from the Bonobo monkey, and transplants it into the womb of, say, a Chimpanzee. The scientist then studies the development and future performance of the child, developed in the Chimp womb. Similarly, we select genomes of trusses evolved in some environment, then allow them to re-develop in a different environment, and observe the results in terms of maximization of a specified fitness function.

The re-growth experiments take two forms: Firstly, we consider the re-growth of genomes under different phenotypic sizes. That is, we take a genome evolved under the assumption that growth would continue until a height of 30 m, and re-grow the truss to a height of 40 m; Secondly, we re-grow genomes in different environments altogether.

The purpose of the re-growth experiments is to measure the utility of genomes when translated into phenotypes under perturbed environments; This is done for two reasons: Firstly, to explore the possibility that good developmental patterns exist in several contexts, somewhat like the genetic toolkit currently being explored in real-world Embryology; Secondly, to reinforce the claim that Artificial Embryogeny is a robust and perturbation-resistant technique for automated design, and that good genomes may be re-used in new environments without repeating the evolutionary optimization.

The final experiments were relatively successful; Re-growth using new phenotypic sizes showed great utility, as re-grown trusses far outperformed random genomes, and approached the utility of the trusses grown from genomes evolved at the phenotypic size natively. Re-growth using new environments showed some of the same trends as the preceding experiments, but not nearly so universally. However, in the course of the re-growth experiments, several agents capable of growing high-fitness trusses in nearly *all* explored environments were found. This suggests the existence of a "genetic toolkit" for the Deva 1.N model.

2. REVIEW

We review three relevant fields: Artificial Embryogeny (**AE**); a simple model of structural form, the Plane Truss; and the evolution of structural form through Evolutionary Computation (**EC**).

2.1 Artificial Embryogeny

There is much interest at present in the use of development in Evolutionary Computation. Artificial Embryogeny (sometimes also "Computational Development" and "Artificial Ontogeny") is a term used to describe a developmental phase in artificial evolution, that is, an indirect mapping between the genotype (representation) and phenotype (evaluated organism). It is common, although not necessary, that this mapping be inspired by biological embryogenesis.

The first computational models of embryogenesis include chemical diffusion work by Turing [19] and work with simple automata by Lindenmayer [10]. Much current research revolves around the attempt to reverse-engineer Evo Devo, or to create "plausible" models of embryogeny. These include attempts to model plant growth by Prusinkiewicz *et al*, [13], [14]; to model the expression of genes with cis-regulatory regions, as in Kumar and Bentley's Evolutionary Development System [9]; or to model the environment in which cells grow, as in Eggenberger Hotz' three-dimensional structures [4].

Most relevant to our current interest are cases where AE has been applied to the design of solutions to problems from engineering and related fields. AE possess several attractive properties which imply its potential use in situations where direct encoding might be impossible or intractable. AE techniques are believed to be capable of exploiting a canalization of development, allowing for the design of organisms too large for evolution via bijective encodings [7], [3]. AE is believed to be a mechanism by which large complex systems may maintain themselves, executing self-repair following damage [12]. AE allows for significant environmental influence on the development of organisms, allowing for the same representations to be used in the development of several different organisms [7]. Finally, it has been suggested that AE might be used to generate not only the final organism, but also a constructive map, detailing a plan for the assembly of the final design [16]. There are many examples of fields where AE has been applied to practical engineering problems: Eggenberger Hotz *et al* have used development to grow neural network architectures of impressive size and complexity [5]; Sekanina and Bidlo used evolution and a developmental algorithm to evolve sorting networks [17]; Stoy and Nagpal use a Cellular-Automata-like technique to allow an undifferentiated mass of components to self-organize into a pre-determined shape [18]; Kowaliw *et al* designed a CA-like system capable of evolving agents capable of surviving in a virtual world [7].

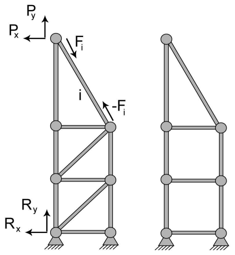

Figure 1: Two plane trusses, the left is stable, the right unstable. Labelled on the left: external force P applied to the top joint, reactive force R from a base joint, member force F_i of the ith member.

2.2 Trusses

Trusses are well studied examples of structural design, being used by architects and engineers in nearly all construction; Often, they are cited as the simplest such model. Still, as an approximation of real-world structures, trusses are close enough to be suitable models for most small construction projects, and are typically used at least in the initial design phase of nearly all large construction. Truss-based structures invisibly form the basis of nearly every large building or tower, but are most obviously visible in bridges, hydro towers, house roofing. Although a simple model, truss design can be exceedingly complex; As such, trusses are an appropriate choice for evaluating a model's ability to perform structural design, allowing for an evaluation of those designs from a completely independent context.

2.2.1 Plane Trusses

Plane trusses are two dimensional constructs consisting of (for our purposes) joints, beams and grounds; An example may be seen in Figure 1. A truss is any connected collection of these three components, regardless of usefulness or triviality. All beams are connected via joints, which may be connected to grounds. The typical purpose of a truss is to support other structures, and to re-distribute any external forces so as to retain its original form.

Given some truss, our natural first questions is whether or not it is stable — i.e., will it (approximately) retain its shape. The second question involves the stress placed on the members under some external force — if the maximum stress exceeds the yield strength of any particular beam, the truss may quickly become unstable. Another important issue involves the deformation of the truss members under strain; Given some beam and an external force, a beam will either compress or stretch, which in turn will cause the truss' joints to dislocate. Figure 1 shows two trusses; The first is stable, but the second is not — any external force would cause the second to deform drastically. We will assume, for all future discussions, that our trusses are topologically connected, pin-connected, friction-free, and that force is applied only at joints.

2.2.2 *Truss Stress Analysis*

We now examine the computation of member-forces in an arbitrary plane truss[1]. There exist some simple counting tests that may determine if a given truss is unstable. Failing that, we must attempt to compute the equilibrium state given some external forces — in the process, we obtain values for all member forces. In our example, all truss members are identical in terms of material and area, grown in a developmental space where units are measured in meters; We specify material and area by setting $EA = 1.57 \times 10^4$ N, corresponding to a modulus of elasticity E for steel [11] and a cylindrical member of diameter 1 cm.

Consider a general truss with n joints and m beams; We are provided with external forces to be applied at joints, and wish to determine the member forces. Let our structure forces be $\{P\} = \{P^1, ..., P^n\}^T$, structure displacements be $\{\Delta\} = \{\Delta^1, ..., \Delta^n\}^T$ and member forces be $\{F\} = \{F^1, ..., F^m\}^T$. We may relate the individual member forces to displacement and structure forces as follows:

$$\{F\}^i = [k]_a^i[\beta]^i\{\Delta\} \qquad (1)$$

where $[\beta]^i$ is the connectivity matrix for the ith member beam, and $[k]_a^i$ is its stiffness matrix, relating the deformation of the beam under a given force to the displacement at the joint. Hence, to solve for forces, it suffices to compute the displacements. The displacements may be computed through a truss stiffness matrix, a combination of the individual member stiffness matrices:

$$\{\Delta\} = [K]^{-1}\{P\} \qquad (2)$$

Hence, given a plane truss, we may first compute the stiffness matrix, then compute the displacements, then the individual member forces. The entire process is bounded by the calculation of a matrix inversion (or LU-Decomposition), and hence has running time $O(m^3)$.

2.3 Evolution of Structures

There has been significant interest in the evolution of structural designs. This has included several frameworks for their analysis, including plane and space trusses, simplified models of Lego, and others. The Lego and related simple models have led to some interesting research in design, including the early development of buildable structures [2], or, more recently, the use of AE for the design of a simple arch, including scaffolding [16]. However, since we desire a notion of structural design which may be evaluated through means external to the A-Life community, we will instead concentrate on models taken directly from Engineering. An extensive recent review was conducted by Kicinger *et al* [6]. Typically, use of EC in structural design concentrates on optimizing the sizing or shape of existing frameworks — our work, however, involves topological design. Use of a GA to optimize a topological design through a relatively bijective relation between genotype and phenotype has been conducted by Rajan [15] (who also optimized sizing and shape). A more complex approach was undertaken by Yang and Soh, who used a GP approach to optimize topology in the context of tall buildings [21].

3. THE MODEL

In this section we describe the Deva 1.N model, then an interpretation of developed organisms as Plane Trusses. Finally, we detail the evolutionary engine.

3.1 The Deva 1.N Model

Let us consider a model[2] which consists of a *developmental space*, D, a collection of *cell types* (or *colours*), C, a set of *actions*, A, and a *transition function*, ϕ. The developmental space, here a subset of $\mathbb{Z}^2$, is a space in which we may grow an organism, endowed with a discrete time. Each point in the lattice is a cell, possibly the empty cell — each non-empty cell may be viewed as an independent agent. Cells change in time by executing one of several actions; Which action is executed is determined by the cell's genome, the transition function.

We now describe the process of growth: developmental space is initialized empty everywhere, save at a central point, which is initialized with a cell of type "1". At every time step, any non-empty cell examines its neighbourhood, and selects an action through the consultation of the transition function. If the cell has sufficient resources (measured via an internal counter, r_c), and has sufficient age, that action is executed. Through this process, the developmental space changes in time — termination occurs when the space is identical to the space that preceded it (guaranteed to occur due to a finite maximum value of r_c). This process may be written more explicitly as:

```
Time $t \leftarrow 0$
Initialize developmental space $D_t$
while $D_t \neq D_{t-1}$ do
  $t \leftarrow t + 1$
  $D_t \leftarrow D_{t-1}$
  for all Cell $c \in D_{t-1}$ do
    if $c$ has sufficient age and $c_{r_c}$ then
      Action $a \leftarrow \phi(\mu_c)$
      Decrement $c_{r_c}$ appropriately for $a$
      Execute $a$ in $D_t$
    end if
  end for
end while
```

A Deva 1.N transition function is a listing of descriptions of possible neighbourhoods of a specified length, $|\phi|$. These rules are tuples of the form $(c, h_1, ..., h_{n_c}, a)$, where c is a colour, $n_c = |C|$ is the number of cell types, a is an action, and h_i is a count of the number of neighbours of cell type i, or a *hormone-level*. Hence, the size of the representation of

[1] This analysis is taken largely from West's treatment [20].

[2] The Deva 1.N model is very similar to the Deva 1 model described earlier [8] — technical details are omitted, the interested reader is urged to consult that source.

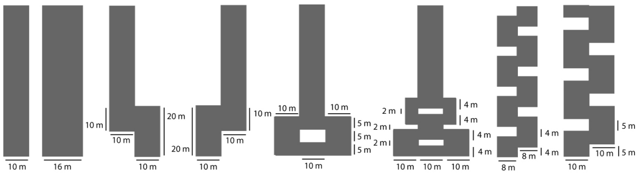

Figure 2: Environments *(left to right)*: thin, thick, swerve-l, swerve-r, bulb, bulbs, zigzag-s, zigzag-l.

such a transition function is $O(|\phi| \cdot n_c)$, and the number of possible transition functions is $n_c \cdot |\mu|^{n_c} \cdot |A|$, where A is the set of all actions and $|\mu| = 12$, the size of a neighbourhood.

Given a current description of a cell and its neighbourhood, the transition function may be queried for an action. Each rule in the transition function is compared to the description of the current neighbourhood, the closest matching rule defined via Euclidean distance. The action associated with the closest rule is returned, or, if there is no matching rule, the "nothing" action. The running time of a transition function lookup is hence $O(|\phi|)$.

Cell actions are the sole means through which the developmental space changes in time. The possible actions are "divide", "die", "elongate", "specialize(x)" and "nothing", where x is a cell colour, and the direction is always in the *best free location*. The best free location is defined to be the empty adjacent location which lies opposite to the greatest mass of non-empty cells (In the case of a tie, we select the left-most location, then clockwise). Most cell actions come with a cost, decrementing a cell's r_c — this is meant to incorporate the notion of finite resources. If a cell cannot execute an action (no best free location, insufficient resources), it does nothing.

A Deva 1.N growth is controlled then through a genome (transition function), several system parameters (number of cell types, n_c, initial setting of resource counter, r_c), and choice of environment.

3.2 Developmental Environments

Each environment is a connected subset of $\mathbb{Z}^2$, where the initial cell is placed in the lowest central location of the space. We use several environments for growth:

- *thin* and *thick*, environments of width 10 m and 16 m, respectively, each pointing straight up.
- *swerve-l* and *swerve-r*, environments of width 10 m which, at height 10 m, swerve left or right, continuing straight up indefinitely.
- *zigzag-l* and *zigzag-s*, a zig-zagging line of width 10 m and 8 m, with corners at heights of multiples of 5 m and 4 m, respectively ("l" and "s" for large and small).
- *bulb* and *bulbs*, environments composed of one or two bulbs, respectively, followed by a shaft of width 10 and unlimited height.
- *normal*, an environment large enough to contain growth.

An illustration of the environments may be seen in Figure 2.

3.3 Interpretation as Plane Trusses

Here, we define a means of interpretting a lattice of cells as a Plane Truss.

Firstly, we define a set of cell types — each non-empty cell will contain a joint, and between zero and five beams. The beams will extend in directions π, $3\pi/4$, $\pi/2$, $\pi/4$ and 0, labelled g_0 through g_4 respectively. Conversion from boolean gene values to an integer is accomplished through the following equation:

$$colour = 2^4 g_4 + 2^3 g_3 + 2^2 g_2 + 2^1 g_1 + 2^0 g_0 + 1$$

The zero cell type is reserved for the empty cell, the one value is for a joint with no beams, and all other combinations exist in the set $\{2, ..., 32\}$.

We may also allow cells to be elongated in one direction, by an arbitrary number of cell lengths. For example, a cell of type 9 has an angle of $3\pi/4$ with the x-axis, and a length of $\sqrt{2}$; A single elongation in the y-direction would lead to a length of $\sqrt{5}$, and an angle of $7\pi/8$ with the x-axis.

Hence, given a lattice of grown cell types and elongations, we may map to a (possibly trivial or useless) truss. Any joints located at the bottom of the space are attached to grounds.

Finally, trusses are trimmed. The trimming process serves to: (a) remove obviously unstable sections, such as beams which do not connect to joints at both ends; (b) to remove sections which are not connected to the base of the structure; and (c) to remove redundant joints, replacing them with longer beams. All three of these can be accomplished in a single pass of the un-trimmed truss structure, allowing for processing in $O(n)$ time, where n is the number of beams.

3.4 Initialization and Genetic Operators

As previously mentioned, an organism may be represented by its transition function. The transition function, in turn, may be represented as a series of rules — that is, $|\phi|$ rules, each represented by $2 + n_c$ integers. Hence, a genome is simply a list of integers.

Here we describe the initialization of a transition function rule. The hormones may be initialized through a power-law distribution which favours 0:

$$Pr[X = i \mid 0 \leq i \leq 12] = \frac{1}{\sum_{j=0}^{12} \beta^j} \beta^{12-i}$$

where $\beta = 3.6$ — this guarantees that most rules describe a possible neighbourhood. Actions are chosen with each of the possibilities equally likely, with each colour of specialization also equally likely. Hence, we may generate a random rule by (uniformly) randomly generating an initial rule colour, then generating n_c hormones and one action according to the above distributions. The initial rule in any transition function in the initial population is set to be a "divide" command.

We define genetic operators as follows: In the case of *crossover*, we use a simple single-point crossover, with the tail ends of two parents' genomes swapped. We use two kinds of point mutation: *power-mutation*, which replaces an integer with another selected from the same distribution used for initialization; and *copy-mutation*, which replaces the current integer with another selected randomly from the current genome.

3.5 Evolution

The use of Deva 1.N for the generation of designs is controlled overall via Evolutionary Computation (EC). That is, genomes are mapped to organisms via the Deva 1.N algorithm, and the organisms are assigned fitness through the truss interpretation. The fitness serves to select a set of genomes for the next generation, and the actual selection and recombination is controlled through a Genetic Algorithm (GA). We use a typical GA, as described by Eiben and Smith [1]. The GA uses elitism, as well as crossover and mutation as defined above. Selection is accomplished through a tournament of five population members, using a tournament probability of $p = 0.7$.

The fitness function used in our trials is very similar to that used in the original Deva 1 experiments — the primary difference being that the external load applied to the trusses has been reduced. The evolution of Plane Trusses may be viewed as a multi-objective evaluation; The factors involved, defined for a general truss T, include:

- Selection for height, $h(T) = h/(r_c + 1)$, where h is the raw height of T.

- Selection for minimal material use, where $m \in [0, 1]$ varies linearly between 0 for maximal use of materials and 1 for none.

- Selection for stability, where T is considered stable if the inverse stiffness matrix is non-singular, and if there are no absurd deformations[3]. The stability criterion is then defined as $s(T) = 1$ if T is stable, $s(T) = 1/4$ otherwise.

- Selection for distribution of pressure, $p \in [1/2, 1]$. Having applied some external force, we measure the maximum absolute beam pressure in the truss, M. If pressure has exceeded our yield limit of 165 MPa (approximately 80% the limit of steel), we return $p = 1/2$; Otherwise,

$$p = \frac{1}{2} + \frac{1}{2}\left(\frac{165\text{ MPa} - |M|}{165\text{ MPa}}\right)$$

At every joint, we apply 10 N down and 10 N left, simulating gravity and a mild horizontal force. Additionally, we apply 2500 N down (approximately 2.5 kg) and 500 N right at the highest joint; In the case of several joints, the force is divided evenly between them. Hence, we seek a tall, minimal structure, capable of supporting a large mass at the top, much like a tower supporting some additional structure at the peak. The fitness of a truss T is thus defined as

$$f(T) = h(T) \cdot m(T) \cdot s(T) \cdot p(T) \tag{3}$$

4. EXPERIMENTS AND ANALYSIS

4.1 Environment-based Evolution

80 runs of 100 GA generations were undertaken, grouped together by environment. These runs will be referred to as $r.env.x$, where env is the name of the environment, and $x \in \{0, .., 9\}$ is an index. Parameter settings for the trials are:

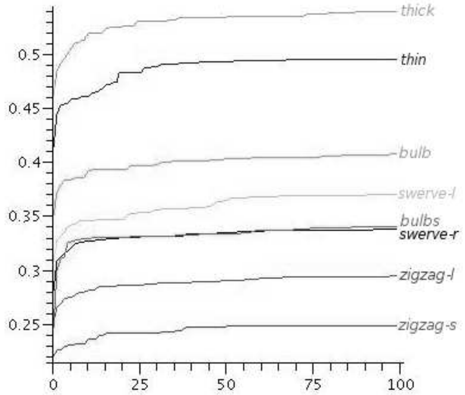

Figure 3: Plot of the mean of maximum fitness for the sets of runs in each environment: fitness (y-axis) versus generation (x-axis).

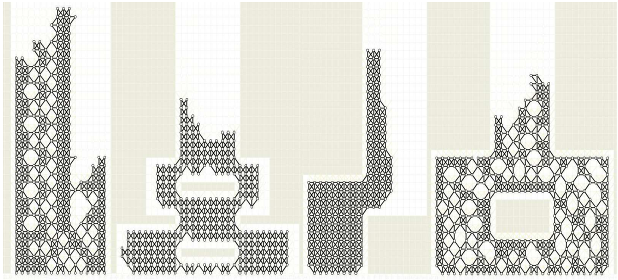

Figure 4: Examples of trusses evolved in the environments: (*left to right*) *thin*, *bulbs*, *swerve-r*, and *bulb*.

population size	100	init. pop. size	1000
prob. crossover	0.8	rate. elitism	0.01
prob. copy-mut.	0.05	prob. power-mut.	0.05
r_c	40	$\|\phi\|$	200

Unless otherwise noted, these parameters are used for all experiments, including re-growth.

In all environments, stable agents capable of supporting the external load were found in nearly all trials, save the *zigzag-s* environment, where only one trial yielded a non-trivial truss capable of supporting the external load. The failure of evolution of a tall, load-bearing truss in the *zigzag-s* environment is a mild disappointment, since it is possible (for a human engineer) to design a truss capable of reaching non-trivial height for this environment. The relative success of the GA, as measured by maximum fitness, varied substantially between runs; A plot of the mean of the maximum fitness for each set of runs may be seen in Figure 3. Some examples of trusses from these runs are shown in Figure 4.

Speaking informally, there seems to exist an inverse relation between the maximum fitness achieved and both of: (a) the complexity of the environment (as measured by, say, number of corners), and (b) the amount of space in the path for growth. For instance, both *thick* and *thin* do well due to both a wide path and low complexity — the failure of *zigzag-s* is likely due to both high complexity, and a too-narrow path vertically (only three meters of space for vertical connections).

[3] Where absurdity kicks in at ten meters or more; This is necessary as the equilibrium process may sometimes find stable points through profoundly unrealistic stretching of materials.

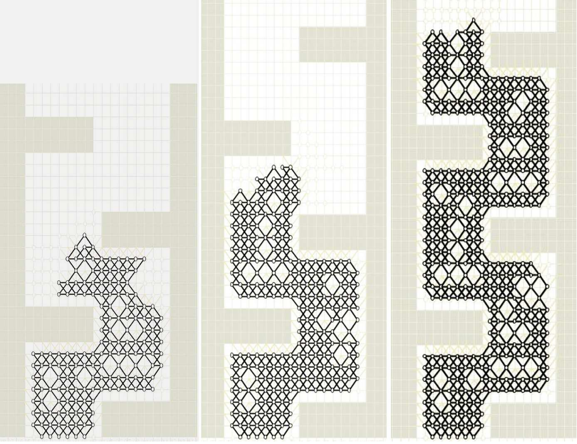

Figure 5: Re-growth of the maximum fit genome from generation 100 of the run *run.zigzag-l-30.3* with (*left to right*) $r_c = 30$ (original), $r_c = 40$ and $r_c = 60$.

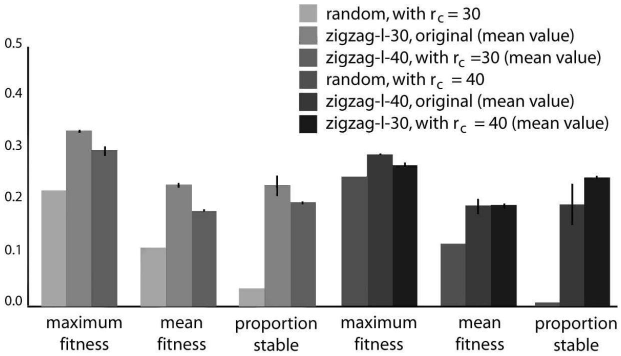

Figure 6: Selected statistics from random genomes and the *r.zigzag-l-30* and *r.zigzag-l-40* runs, both originals and re-growth at $r_c = 30$, 40.

4.2 Re-growth at Different Sizes

An additional set of runs of the *zigzag-l* environment were undertaken, ten with $r_c = 40$, and an additional ten with $r_c = 30$; We shall refer to these sets as *r.zigzag-l-30* and *r.zigzag-s-40*. These sets afford us opportunity to view the re-growth of populations of agents at a different size of environment that the one in which they were evolved. Each of the populations from the above runs were re-grown using the alternate value of r_c; By re-grown, we mean that the population at GA generation 100 was re-developed and re-evaluated using the new value of r_c. The results are summarized in Figure 6; Additionally, data for the expected performance of randomly generated genomes (over 2000 genomes) are also plotted. An example of the re-growth of the maximum fit genome from generation 100 of run *r.zigzag-l-30.3* is shown in Figure 5.

Is it evident from Figure 6 that genomes evolved in a different size perform better, in terms of fitness and proportion of stable agents, than random genomes — the comparison for the $r_c = 30$ *zigzag-l* trials was an expected maximum fitness of 0.225 versus 0.303, or 135% performance for the latter (these values encompassing the difference between finding a load-bearing truss and not); Similar results are true for mean and proportion in both the $r_c = 30$ and $r_c = 40$ experiments. A more meaningful comparison is between the evaluation of genomes evolved at the present size, and those transplanted from a different size. In the $r_c = 30$ trials, the original agents have an expected maximum fitness of approximately 0.341 versus 0.303 for the transplanted agents, meaning 113% the performance for the original. In the $r_c = 40$ trials, results are much closer: an expected maximum fitness of 0.293 for the original, versus 0.277 for the transplanted genomes, or 105% the performance for the original. Indeed, in the $r_c = 40$ case, the mean fitness of proportion of stable agents was higher for the transplanted genomes than the originals.

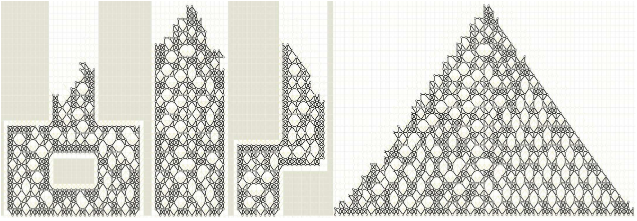

Figure 7: Re-growth of the maximum fit genome from generation 100 of the run *run.bulb.3* in environments (*left to right*) *bulb* (original), *thick*, *swerve-r*, *nothing*.

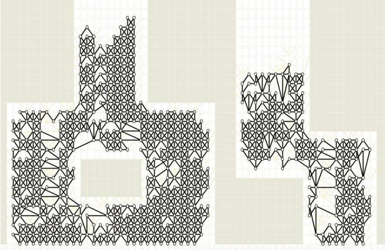

Figure 8: Growth of the maximum fit genome from generation 100 of the run *run.bulb.1* in (*left*) its original environment and (*right*) re-grown in the *swerve-l* environment.

4.3 Re-growth in Different Environments

For the *r.thick* and *r.bulb* runs from Section 4.1, the maximum fitness population (at GA generation 100) was chosen for further experimentation in different environments; These two particular environments were chosen as examples of a simple and a complex environment, by the informal standard of number of corners. The re-growth experiments were accomplished by re-developing the population members using the same parameters, but a different environment. Mean data from these runs is summarized in Figure 9, compared to both the maximum fitness of a set of 2000 random genomes, and against the mean maximum fitness obtained from the initial runs in Section 4.1. A more detailed view of the data may be seen in Tables 1 and 2.

Performance of the first set, the *r.thick* trials, showed some improvement over the maximum fitness of the random set of agents in most environments. In five out of eight environments, the fitness of re-grown genomes exceeded that of the best of the set of 2000 random genomes. Performance

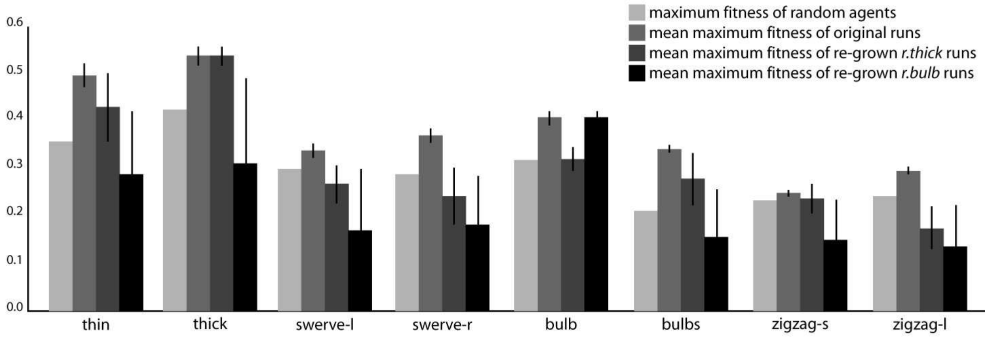

Figure 9: Selected statistics from random genomes and the *r.thick* and *r.bulb* runs, both originals and regrowth in a variety of environments

Table 1: Maximum fitness from the re-growth of the *run.thick* trials (maximum fit agent at generation 100) in different environments.

orig. pop.	thin	thick	swerve-l	swerve-r	bulb	bulbs	zigzag-s	zigzag-l	nothing
r.thick.0	0.463	0.545	0.358	0.279	0.344	0.327	0.176	0.255	0.073
r.thick.1	0.399	0.557	0.311	0.281	0.281	0.323	0.224	0.136	0.509
r.thick.2	0.460	0.522	0.232	0.279	0.345	0.342	0.240	0.206	0.525
r.thick.3	0.524	0.536	0.234	0.234	0.290	0.251	0.263	0.196	0.073
r.thick.4	0.468	0.567	0.231	0.231	0.341	0.303	0.241	0.114	0.073
r.thick.5	0.300	0.507	0.273	0.241	0.335	0.313	0.248	0.195	0.514
r.thick.6	0.510	0.548	0.235	0.319	0.292	0.179	0.263	0.115	0.544
r.thick.7	0.495	0.567	0.238	0.239	0.297	0.302	0.193	0.123	0.073
r.thick.8	0.384	0.520	0.313	0.074	0.338	0.275	0.282	0.189	0.502
r.thick.9	0.299	0.507	0.256	0.240	0.335	0.178	0.235	0.208	0.528

Table 2: Maximum fitness from the re-growth of the *run.bulb* trials (maximum fit agent at generation 100) in different environments.

orig. pop.	thin	thick	swerve-l	swerve-r	bulb	bulbs	zigzag-s	zigzag-l	nothing
r.bulb.0	0.345	0.489	0.049	0.072	0.403	0.052	0.134	0.052	0.544
r.bulb.1	0.090	0.076	0.053	0.072	0.404	0.043	0.061	0.052	0.073
r.bulb.2	0.442	0.493	0.324	0.293	0.424	0.342	0.048	0.253	0.552
r.bulb.3	0.350	0.457	0.055	0.293	0.412	0.049	0.064	0.055	0.555
r.bulb.4	0.349	0.071	0.049	0.072	0.395	0.199	0.249	0.053	0.545
r.bulb.5	0.357	0.470	0.226	0.074	0.422	0.051	0.255	0.048	0.562
r.bulb.6	0.083	0.074	0.057	0.297	0.395	0.297	0.060	0.187	0.071
r.bulb.7	0.086	0.485	0.225	0.283	0.408	0.044	0.245	0.218	0.544
r.bulb.8	0.347	0.071	0.336	0.072	0.389	0.259	0.248	0.214	0.545
r.bulb.9	0.433	0.420	0.326	0.290	0.424	0.225	0.134	0.227	0.559

of the second set, the *r.bulb* trials, was more disappointing overall, with most mean values falling below the maximum fitness found during the search of random genomes. This is due largely to several genomes, particularly those from runs with indices 1, 2 and 4, who performed very poorly in any foreign environment — likely, the relative complexity of the *bulb* environment invites over-specialization during evolution. One such unsuccessful re-growth is shown in Figure 8. However, there exist several genomes which have performed admirably in all environments, in some cases exceeding the mean maximum fitness of the original sets; One such example is illustrated in Figure 7.

5. CONCLUSIONS

The primary purpose of these experiments has been to show that environment may be used as a spatial constraint in the design of structural form, using Artificial Embryogeny. Indeed, for several diverse and non-trivial environments, this has been shown to be the case.

Further, we explored, through additional experiments, the possibility of re-use of genomes from a particular environment in different settings. This is both a means of evaluating the general claim that AE growth is robust and resistant to environmental perturbations, and also a means of demonstrating that AE genomes may be re-used in different contexts without re-running the evolutionary process.

The experiments involving re-growth at different values of r_c showed that, in the *zigzag-l* environment, the fitness of re-grown agents far exceeds that of random genomes, and approaches the fitness of genomes evolved at the new size in question. Although we must hesitate before extending these results to all sizes and environments, it seems likely that genomes evolved at some particular size have utility when re-grown at different sizes.

The re-growth experiments involving different environments were less successful as those involving phenotypic size; This was largely due to several genomes which failed to perform well in any environment other than the one in which they evolved, suggesting a measure of over-specialization. In these final experiments, however, a set of genomes capable of developing into high fitness trusses in nearly any environment were found; These latter agents suggest the existence of a general toolkit which might be useful for a wide class of AE design problems. Analysis of these genomes may, in the future, help to design better algorithms for AE.

The experiments in re-growth generally raise the possibility that genomes evolved via Artificial Embryogeny may be re-used in slightly different contexts, notably contexts of different phenotypic size or different environment, without significant re-design; This strengthens the claim that growth through Artificial Embryogeny in general is a robust and perturbation-resistant means of automated design.

6. REFERENCES

[1] A. E. Eiben and J. E. Smith. *Introduction to Evolutionary Computing*. Springer, 2003.

[2] P. Funes and J. Pollack. Computer evolution of buildable objects. In P. Husbands and I. Harvey, editors, *Fourth European Conference on Artificial Life*, pages 358–367, 1997.

[3] S. Harding and J. Miller. The dead state: A comparison between direct and developmental encodings. In *GECCO*, 2006.

[4] P. E. Hotz. Combining developmental processes and their physics in an artificial evolutionary system to evolve shapes. In S. Kumar and P. Bentley, editors, *On Growth, Form and Computers*, pages 302–318. Elsevier Academic Press, 2003.

[5] P. E. Hotz, G. Gomez, and R. Pfeifer. Evolving the morphology of a neural network for controlling a foveating retina and its test on a real robot. In *Artificial Life VIII: 8th Int. Conf. on the Simulation and Synthesis of Living Systems*, 2003.

[6] R. Kicinger, T. Arciszewski, and K. A. D. Jong. Evolutionary computation and structural design: a survey of the state of the art. *Computers & Structures*, 83(23-24):1943–1978, 2005.

[7] T. Kowaliw, P. Grogono, and N. Kharma. Bluenome: A novel developmental model of artificial morphogenesis. In *GECCO*, 2004.

[8] T. Kowaliw, P. Grogono, and N. Kharma. The evolution of structural design through artificial embryogeny. In *Proceedings of the IEEE First International Symposium on Artificial Life*, 2007. FORTHCOMING.

[9] S. Kumar and P. Bentley. *On Growth, Form and Computers*. Elsevier Academic Press, 2003.

[10] A. Lindenmayer. Mathematical models for cellular interaction in development. *Journal of Theoretical Biology*, 18:280–315, 1968.

[11] T. H. G. Megson. *Structural and Stress Analysis, 2nd Ed.* Elsevier Butterworth Heinmann, 2005.

[12] J. Miller. Evolving a self-repairing, self-regulating, french flag organism. In *GECCO*, 2004.

[13] P. Prusinkiewicz and A. Lindenmayer. *The Algorithmic Beauty of Plants*. Springer-Verlag, 1990.

[14] P. Prusinkiewicz and A. Rolland-Lagan. Modeling plant morphogenesis. *Current Opinion in Plant Biology*, 9:83–88, 2006.

[15] S. D. Rajan. Sizing, shape and topology design optimization of trusses using a genetic algorithm. *Journal of Structural Engineering*, 121:1480–1487, 1995.

[16] J. Rieffel and J. Pollack. The emergence of ontogenic scaffolding in a stochastic development environment. In *GECCO*, 2004.

[17] L. Sekanina and M. Bidlo. Evolutionary design of arbitrarily large sorting networks using development. *Genetic Programming and Evolvable Machines*, 6(3):319–347, 2005.

[18] K. Stoy and R. Nagpal. Self-reconfiguration using directed growth. In *Proc. 7th Int. Symp. on Distributed Autonomous Robotic Systems*, pages 1–10, 2004.

[19] A. Turing. The chemical basis of morphogenesis. *Philosophical Transactions of the Royal Society B*, 237:37–72, 1952.

[20] H. H. West. *Analysis of Structures: An Integration of Classical and Modern Methods*. John Wi,ley and Sons, 1989.

[21] Y. Yang and C. K. Soh. Automated optimum design of structures using genetic programming. *Computers & Structures*, 80(18-19):1537–1546, 2002.