An Evo-Devo Approach to Architectural Design

Daniel Richards
Manchester Institute for Research and Innovation in Art & Design
Manchester Metropolitan University
Manchester M15 6BG, UK
D.Richards@mmu.ac.uk

Nick Dunn
Manchester Institute for Research and Innovation in Art & Design
Manchester Metropolitan University
Manchester M15 6BG, UK
N.Dunn@mmu.ac.uk

Martyn Amos
School of Computing, Mathematics & Digital Technology
Manchester Metropolitan University
Manchester M1 5GD, UK
M.Amos@mmu.ac.uk

ABSTRACT
We present a developmental genotype-phenotype growth process, or embryogeny, which is used to evolve, *in silico*, efficient three-dimensional structures that exhibit real-world architectural performance. The embryogeny defines a sequential assembly of architectural components within a three-dimensional volume, and indirectly establishes a regulatory network of components based on the principles of gene regulation. The implicitly regulated phenotypes suggest advances for the automatic design of physical structures, by improving scalability of the genotype encoding and embedding real-world constraints. We demonstrate that our model can evolve novel, yet efficient, architectural structures which exhibit emergent shape, topology and material distribution. Finally, we compare evolved structures against a "hand-coded" solution to illustrate that our model produces competitive results without prior knowledge of the design solution or direct human guidance.

Categories and Subject Descriptors
I.2.11 **[Distributed Artificial Intelligence]**; J.5 **[Arts and Humanities]**: Architecture; J.6 [**Computer-aided Engineering]**: Computer-aided Design (CAD).

General Terms
Algorithms, Performance, Design, Experimentation

Keywords
Artificial Embryogeny, Morphogenetic Engineering, Architectural Design, Self-Organization, Computational Design Synthesis.

1. INTRODUCTION
The physical properties of natural morphologies are extraordinary. Through complex processes of self-organization and evolution, nature is able to assemble extremely efficient material structures that are economical, multifunctional and exhibit magnitudes of complexity that greatly surpass anything we can currently design. Architects and engineers have often taken inspiration from nature; today, in the face of rapid climate change and urbanization, they seek new bio-inspired methods of designing "sustainable" performance-oriented architectural form.

Traditional "form-first" design approaches often consider real-world performance, such as structural integrity or solar gains, late in the design process. Consequently, the structures produced are often wasteful in their use of material and composed of many mono-functional sub-systems which render them far less "sustainable" and "integrated" than the complex material structures found in nature. An emerging alternative design paradigm, often termed computational design synthesis, seeks to invert the traditional approach [1-4]. Our proposed method extends this field of research by using bio-inspired computation to transform physical parameters from *passive* attributes into *active* properties that shape development. Rather than being ascribed to form post-design, features such as material properties, fabrication constraints and environmental performance are allowed to influence the early development and synthesis of form. Our ultimate goal is *automatic self-organization* of real-world components into efficient architectural morphologies. We use an evolutionary algorithm (EA) with a novel developmental genotype-phenotype encoding to construct highly integrated three-dimensional structures that exhibit competitive real-world performance, and express emergent shape, topology and material distribution.

Computational design synthesis research has demonstrated *semi-automated* models that use performance simulations to inform the early development of form [1,2,5]. However, these methods are limited to addressing relatively simple design problems, since they require the manual design, modification and interpretation of *parameterized design representations* in response to performance analysis. Parameterized representations have proved successful when applied to late-stage optimization, but they assume a prior understanding of the design problem, so that key parameters can be identified and encoded before computation is applied. To progress existing research and eventually address complex design problems (where prior understanding of the solution is unavailable) we require *open-ended design representations* [6,7] that allow the topology of solutions to be modified throughout an evolutionary "design" process. Existing application of open-ended representations within architectural design has been limited, yet it is argued that the ability to integrate simulation and performance analysis with such representations will facilitate the self-organization of extremely integrated, performance-driven structures [8-10].

This paper presents a novel developmental representation that is *explicitly* defined, yet *implicitly* [20] regulated using an indirect genotype-phenotype mapping inspired by genetic regulatory networks. The developmental mapping is combined with a simple

GECCO'12, July 7-11, 2012, Philadelphia, USA.

evolutionary algorithm (EA) to construct an evo-devo model, and used to demonstrate the self-organization of efficient architectural structures without the aid of a designer.

The paper is structured as follows: In Section 2 we review related work that utilizes developmental or generative encodings within EAs to construct performance-oriented structures. In Section 3 we present the genotype encoding and developmental mapping process. Section 4 describes our experimental results, and we conclude in Section 5 with a summary and discussion of further research challenges and opportunities.

2. RELATED WORK

Evolutionary computation has been widely applied to design and engineering for the post-design optimization of parameterized (fixed topology) design representations (see Kicinger, *et al*. for an extensive overview [11]). Existing attempts to integrate EAs into the early stages of design have focused upon either the iterative design and "calibration" of parameterized representations to enhance architectural performance [12-13], or the use of generative, open-ended representations for the creative exploration of formal possibilities [14-16]. However, the successful application of EAs to synthesize *shape, topology* and *material distribution* for the self-organization of highly integrated architectural form has not, until now, been demonstrated. Existing work has demonstrated that open-ended representations can produce *well-performing* structures without the need for *well-defined* parameterized representations. Shea *et al*. use an open-ended representation based on shape grammars to evolve novel truss structures early in the design process [17], and Funes and Pollack demonstrate a grammar-based encoding to evolve structurally robust Lego structures [18]. This research illustrates the successful use of open-ended representations to encode and evolve physical structures; however, these models require a direct genotype-phenotype mapping process, which has proved to be less effective at finding good solutions as design problems increase in scale and complexity [6,18]. To improve scalability, the use of bio-inspired developmental growth processes, or *embryogenies*, has been proposed as a means of indirectly mapping from genotype (encoding) to phenotype (structure) [19-21].

Explicit embryogenies [20], which use indirect grammar-based representations, have been used to evolve three-dimensional structures for architectural design. O'Reilly and Hemberg demonstrate the evolution of grammar-based three-dimensional surfaces within a simulated environment [15]. von Mammel and Jacob present a swarm-based grammar system for evolving architectural forms [22], O'Neil *et al*. use grammatical representations to evolve simple architectural shelter designs [23], and Rieffel *et al*. use a graph-based encoding for finding and optimizing large irregular tensegrity structures [24]. The major limitation of existing explicit embryogenies, when applied to architectural design, is inclusion of real-world performance. Firstly, evolved forms often exhibit "unbuild-able" intersecting geometries, as Sims found in his work on early virtual creatures [25]. Secondly, the inclusion of real-world materiality, construction logic and spatial performance has been limited in existing works, often necessitating evolved forms be fabricated as small scale 3D printed objects [6,26] or remain in silico [22,23].

Implicit embryogenies [20], which utilize bio-inspired regulatory networks to *activate* and *suppress* genes within a developmental representation, have also been used to evolve structures. Eggenberger demonstrates a model of gene regulation that is combined with a genetic algorithm (GA) to evolve "multicellular" three-dimensional shapes [27]. Bongard and Pfeifer use a genetic regulatory model to simultaneously evolve three-dimensional morphologies and neural controls of computational agents within a simulated environment [28]. Miller presents a developmental cellular model to evolve two-dimensional graphics that are able to self-repair when damaged [29]. Hiller and Lipson use a compositional pattern positioning network to evolve amorphous robot morphologies [30] and Yogev *et al.* apply a developmental embryogeny to evolve three-dimensional structures to support simulated structural loadings [31]. The advantages of implicit embryogenies for real-world architectural design problems are twofold. Firstly, implicit embryogenies have been shown to scale efficiently [20], and are therefore suitable for solving complex design problems. This is significant, because existing attempts to bridge the "reality gap" between computational self-organization and real-world fabrication necessitate parameterized or one-to-one open-ended representations [17-18] that are inherently limited to simple design problems. Secondly, real-world design problems are necessarily constrained by fabrication technologies and the physical properties of available components. We suggest that implicit representations can successfully utilize real-world assembly logic, material properties and sizing of components to ensure the development of diverse - yet "build-able" - phenotypes. This can be achieved using developmental regulatory processes inspired by *gene expression, gene repression* and *programmed cell death* (see Section 3).

The developmental design methodology presented in this paper uses an implicitly regulated genotype-phenotype mapping, inspired by genetic regulatory networks, to evolve multifunctional architectural morphologies from a programmable library of architectural components.

3. EMBRYOGENY

The design of our encoding has two principle advantages. Firstly, it allows diverse structures to be represented using small amounts of information. Secondly, it is modular, so can be easily reused to evolve novel libraries of architectural components in response to variable performance criteria. To accomplish this, the genotype encoding takes inspiration from encoding mechanisms in evolutionary developmental biology. The embryogeny defines a *developmental growth process* whereby structural nodes are used to sequentially define a network of material components that constitute a larger architectural structure.

3.1 Genotype Encoding

During the *developmental embryogeny*, characteristics of nodes are used to assemble three-dimensional structures using various material components from the predefined libraries. Within the two predefined libraries there are three types of geometry and nine different classes of material that nodes can use to create material connections. The three types of geometry are: struts between nodes to form branching structures (Fig 1d), struts connecting multiple nodes in a clockwise sequence to form rings (Fig 1e) and surfaces connecting exactly three nodes to form petal-like components (Fig 1f). The nine material classes represent nine different sheet-cut aluminum components which differ in width and cross-sectional area (Fig 2a). Surfaces between nodes are achieved by bending aluminum sheets and connecting at structural seams (Fig 2b). Each material class offers a different trade-off between weight, cost and structural strength, allowing nodes to

construct a diverse array of material connections. Each genotype comprises a string of integers which define the behavior of a population of nodes within a three-dimensional volume. Figure 1 and Figure 2 describe the genotype in detail.

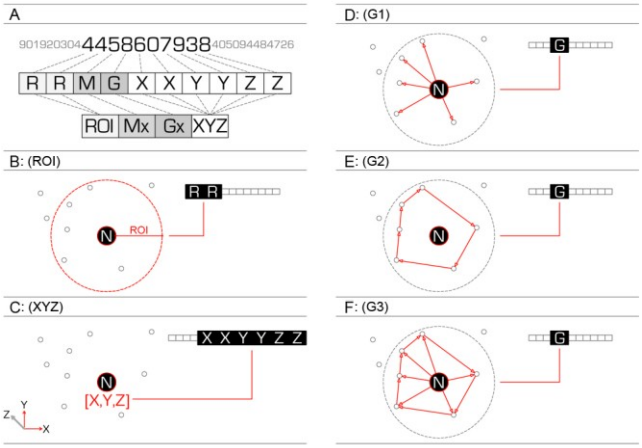

Figure 1. Genotype structure: (A) Each node is described by 4 genes, which define 4 aspects of its development: ROI, M, G and [X,Y,Z]. (B) The *range of influence* (ROI), a radial dimension, described by a 2 digit gene (in the range 0-99), within which nodes can perceive, communicate and connect to other nodes. (C) The *position* of the node within 3D space as described by a 6-digit gene (each digit in the range 0-9) that defines the Cartesian coordinates [XX,YY,ZZ]. (D-F) The library of *geometry types*, G, that the node uses to construct structural connections within its ROI. This is specified using a 1 digit gene (in the range 0-9) to select either: G1,G2 or G3.

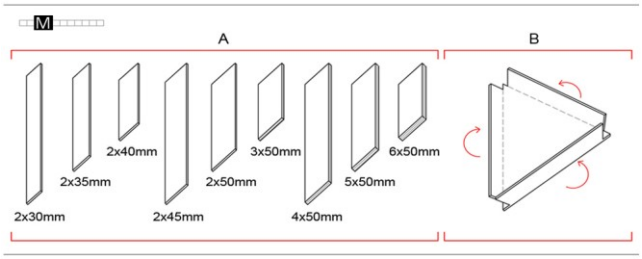

Figure 2. Library of material classes. (A) The *material properties*, M, of connections the node can create. Each nodes' M is defined using a 1 digit gene (in the range 0-9) to select a connection type from the predefined library of components. (B) Surfaces are created by bending aluminum sheets and connecting at structural seams.

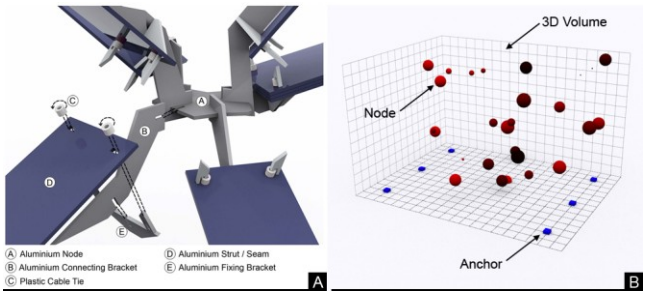

Figure 3. (A) Parametrically defined node detail. (B) The model comprises a collection of nodes within a 3D volume and predefined anchor points.

The length of material connections is controlled by each node's ROI. Nodes which have large ROI values can effectively "see" further, and may subsequently construct longer connections between potentially greater numbers of nodes. All components can be fabricated by cutting 1mm sheet aluminum on a CNC milling machine, therefore the maximum ROI is defined by the maximum dimensions of the CNC mill. Joints between material connections (structural members) are constructed using a *parametrically* defined fixing detail, which may also be cut from sheet aluminum (Fig 3a). In addition to node-node connections, nodes are also able to construct material connections between various anchor points within the three-dimensional volume, which transmit loads directly to a ground surface. Fig 3b illustrates the model set-up with six anchor points defined.

3.2 Developmental Mapping

We now describe the developmental embryogeny to decode the genotype and produce a set of *assembly instructions* that can be used to construct complex three-dimensional structures. In this context, "assembly instructions" simply describe the behavior of individual nodes. Each node operates as a genetic switch that, when activated, constructs local material connections with other nodes within its ROI; thus the local behavior of nodes collectively defines the larger three-dimensional structure. However, as material connections have structural depth and weight, there can only ever be one connection between any two nodes, in order to prevent intersecting geometry. This requires a *sequential growth* of material connections, in a hierarchical fashion defined by node index. This means that the order in which nodes construct material connections is significant, and makes the entire assembly process operate as a primitive genetic regulatory network. Material connections that are constructed by the early growth of nodes indirectly activate or suppress the growth of subsequent connections via processes of *positive* and *negative regulation*.

Within biological systems, positive and negative regulation is observed in the complex processes of gene expression [32]. Put simply, genes are switched "on" or "off" by the existence or non-existence of specific proteins (in reality, genes are rarely binary). If a gene is switched "on", or *expressed*, a protein is produced, which will in turn control the expression or repression of other genes through positive regulation. Conversely, if the gene is switched "off", or *repressed*, no protein is transcribed which influences the expression or repression of other genes, known as negative regulation. Whilst existing work has used biologically analogous models of gene regulation to evolve complex morphologies [27,28], our model makes no such attempt to accurately model genetic regulatory networks. However, it does utilize the basic mechanisms of positive and negative regulation to produce three-dimensional structures that are highly sensitive to nodal behavior via a complex network of regulatory dependencies.

In order for nodes to establish connections and dependencies (topology), each node is given a small *memory space* in which it stores information about other nodes to which it is currently connected. Before each node constructs connections between itself and nodes within its ROI, the node first checks that it does not *already* have an imprint of each node (analogous to a protein) in its memory space (analogous to a cis-regulatory region of DNA). If the node finds an imprint then no connection is made, but if no imprint exists then a new material connection is grown, and each node exchanges a copy of their node index in order to prevent further connections being made.

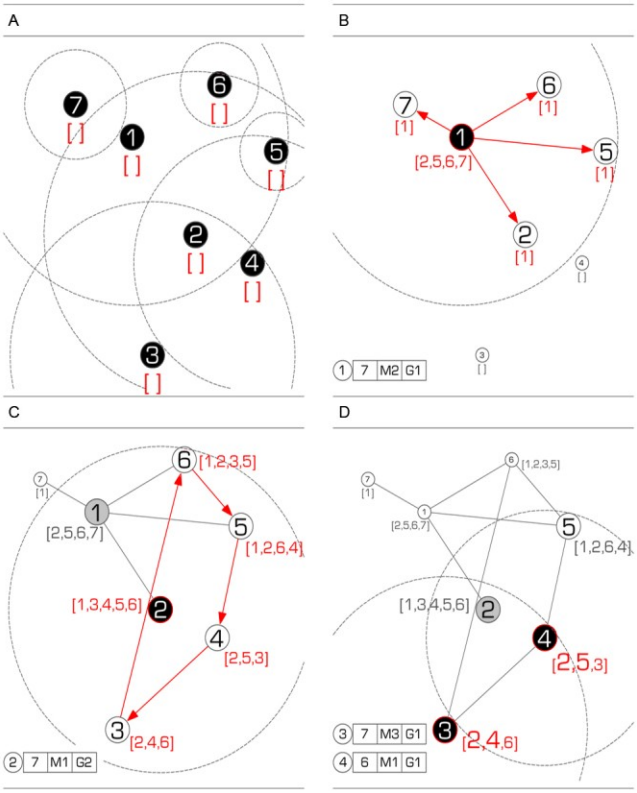

Figure 4. Developmental growth. (A) Shows a random arrangement of nodes with empty memory spaces. (B) Node 1 grows G1 connections between itself and nodes within its ROI. (C) Node 2 grows G2 connections between surrounding nodes – omitting Node 1 as it already has a connection. (D) Nodes 3 and 4 have been repressed by Node 2's earlier growth.

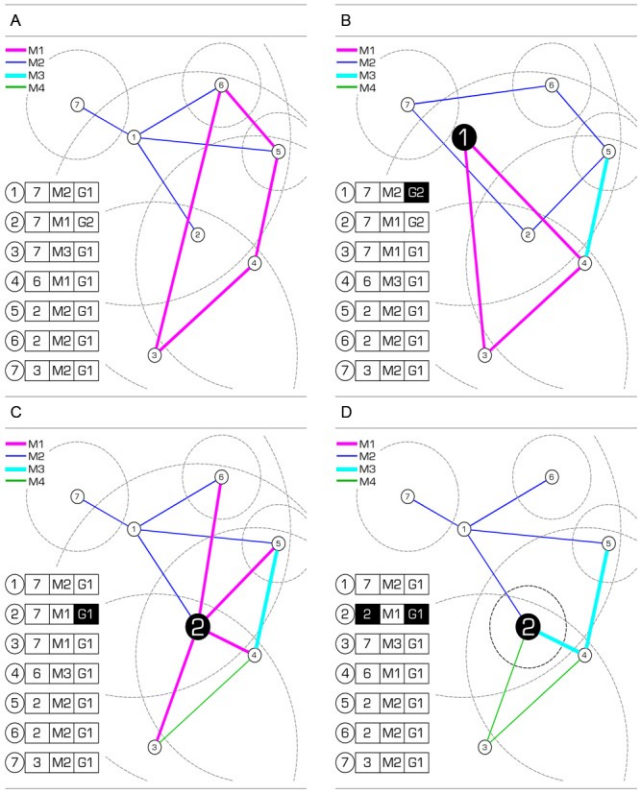

Figure 5. Phenotype variability. Small changes to node ROI or geometry type dramatically influence phenotype formation through positive and negative regulation. (A) Phenotype produced from Fig 4. (B) Mutation of Node 1 geometry type – G1 to G2. (C) Mutation of Node 2 geometry type – G2 to G1. (D) Mutation of Node 2 ROI – ROI = 7 to ROI = 2.

Figure 4 illustrates the developmental growth process using 7 nodes. For clarity, the nodes are set within a two-dimensional space, and each node is defined by 3 genes which control: ROI, material class and geometry type. Each node's memory space is illustrated as square brackets to demonstrate how information is exchanged during assembly. Figure 5 illustrates how small changes to node characteristics provide significant alterations to the phenotype structures.

Following the sequential assembly process, two further operations are performed to remove unbuild-able components from the structure, using a procedure inspired by the biological process of programmed cell death (PCD) [33]. Figure 6 illustrates the process of PCD in detail.

By combining the developmental embryogeny with a standard GA we are able to evolve multifunctional three-dimensional structures that exhibit emergent shape, topology and material distribution. In the next Section, we describe our experimental results.

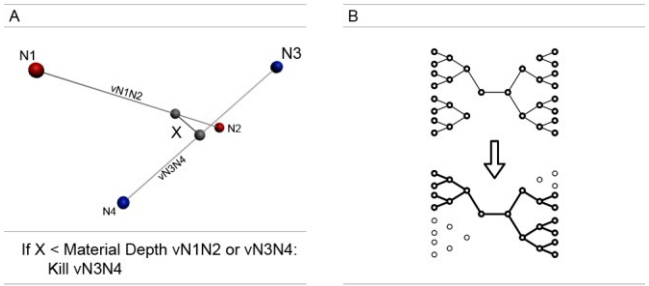

Figure 6. Programmed cell death. (A) Following the developmental growth process, if any components intersect the last component to be formed initiates an automatic suicide program and is deleted. (B) Following the intersection test a connectivity test is used to identify and remove any components which are disconnected from the main structure.

4. EXPERIMENTAL RESULTS

We test our *evo-devo* model on a simple experimental design problem. The problem was to "design" an economical, free-standing structure that can provide controlled (passive) solar shading to a space during the summer months and must be fully fabricated using 1mm CNC cut aluminum sheets. To address this design problem we limited the maximum build volume of structures to 2000 x 1500 x 1100mm and the amount of nodes to 30 based upon initial trials.

We designed two libraries of components that can be fabricated using 1mm aluminum sheeting (Fig 2) and programmed the maximum ROI of nodes to correspond with the maximum cutting dimensions of our CNC mill (840 x 900mm). This approach allows us to represent "build-able" three-dimensional structures with a 120-gene genotype (300 digits in the range 0 9) using our developmental embryogeny.

To provide a baseline for comparison, we hand-designed a solution, shown in Fig 7.

4.1 Fitness Function

The fitness of each phenotype is calculated using the weighted sum of four performance measures: Firstly, we considered the *average nodal deflection* of structures under loading. Secondly, we measured the *average height* of structural nodes. Thirdly, *total fabrication cost* was considered, based on material usage, and finally, we used a measure of daily *solar performance*.

Structural fitness (StF) is calculated using physical simulation (see Implementation). The average combined deflection in [x,y,z] position of structural nodes (AvD) is compared with an acceptable nodal deflection value (AcD) and an unacceptable deflection value (UaD) that defines the breaking point of the component when subjected to a combination of dead weight of structure and an imposed 0.2KN wind loading (1):

$$\mathrm{StF} = 1 - \left(\frac{\mathrm{AvD} - \mathrm{AcD}}{\mathrm{UaD} - \mathrm{AcD}} \right) \tag{1}$$

$$\text{If } \mathrm{StF} > 1 : \mathrm{StF} = 1, \text{ If } \mathrm{StF} < 0 : \mathrm{StF} = 0$$

Height fitness (HeF) is defined by the average height of structural nodes (AvH) in relation to the maximum build volume height (BvH). In this experiment, tall models are favored in an attempt to consider how accommodation requirements of structures might be evolved in larger models. (2):

$$\mathrm{HeF} = \frac{\mathrm{AvH}}{\mathrm{BvH}} \tag{2}$$

The total fabrication cost (CoF) is defined by the required amount of aluminum sheeting (ReA) to fabricate the structure. This is compared against a defined fabrication budget (FaB) and a maximum overspend allowance (MoS). For illustrative purposes the FaB = £300 and MoS = £500 for this experiment (3):

$$\mathrm{CoF} = 1 - \left(\frac{\mathrm{ReA} - \mathrm{FaB}}{\mathrm{MoS} - \mathrm{FaB}} \right) \tag{3}$$

$$\text{If } \mathrm{CoF} > 1 : \mathrm{CoF} = 1, \text{ If } \mathrm{CoF} < 0 : \mathrm{CoF} = 0$$

Solar fitness (SoF) is defined by the average differentiation (AvDiff) of each solar analysis grid cell and a desired measure of solar performance (DeS). Full exposure of any grid cell during analysis returns 2400 KWh. For illustrative purposes DeS = 1500KWh in this experiment (4):

$$\mathrm{SoF} = 1 - \left(\frac{\mathrm{AvDiff} - \mathrm{DeS}}{2400 - \mathrm{DeS}} \right) \tag{4}$$

$$\text{If } \mathrm{SoF} < 0 : \mathrm{SoF} = 0$$

Finally, the overall fitness of each phenotype (F) is calculated by the weighted sum of StF, HeF, CoF and SoF. The relative weightings correspond to the importance of each performance-based attribute in the architectural solution (5):

$$\mathrm{F} = \left(\frac{\mathrm{StF} \cdot \mathrm{w1} + \mathrm{HeF} \cdot \mathrm{w2} + \mathrm{CoF} \cdot \mathrm{w3} + \mathrm{SoF} \cdot \mathrm{w1}}{4.5} \right) \tag{5}$$

$$\mathrm{w1} = 1.5 ; \mathrm{w2} = 1; \mathrm{w3} = 0.5$$

4.2 Implementation

The model is implemented with Python on a 2.7GHz 8-core Windows OS with 6GB RAM, using a combination of COM, OLE and DDE servers to tie together three commercial CAD packages: *3DS Max, Multiframe 3D* and *Ecotect.*

3DS Max is used to create three-dimensional geometry and construction details via a COM server using the inbuilt scripting language *Maxscript.*

Multiframe 3D is used to perform structural analysis of structures using finite element analysis (FEA). Structural data is input using an OLE server and used to accurately define the material properties and imposed physical forces upon phenotypes. Nodes are considered as rigid joints during analysis, whereas anchor points and nodes with a height value of zero are defined as fixed to the ground surface. Results of the FEA return the total deflection for each node in response to the dead weight of the structural components and optional live loadings that can be imposed on all nodes and structural members by a designer.

Ecotect is used to perform basic solar analysis using a .dxf model exported from 3DS Max, and is operated by python using a DDE server and the inbuilt scripting language *Lua*. An analysis grid is placed on the ground plane of the defined volume, and each grid cell measures how much solar radiation it is exposed to over a specified time period. In these experiments, the solar duration is between the hours of 10:00 to 14:00 from 1st June to 31st August and the location is Manchester, UK. Results of the solar analysis return the average daily solar radiation experienced at each cell, and is used to determine the deviation from an idealized (uniform) radiation value that is defined in the fitness function of the GA.

4.3 GA Parameters

Population size = 100, generations = 60, runs = 10, selection type = tournament, tournament size = 5, crossover type = one-point, crossover rate = 100% and mutation rate = 0.005%.

4.4 Results

We now present the results obtained over 10 runs of the GA and compare them against the human designed, "hand coded" solution. The hand coded solution uses the same material and construction details as the evolved models, and was designed using a traditional *form-first* approach. A simple arch was proposed using 6x50mm aluminum members, and then clad using non-structural aluminum panels. Each panel has a different sized aperture. South facing panels have small apertures to allow shading, whereas north facing panels have larger apertures to reduce material costs.

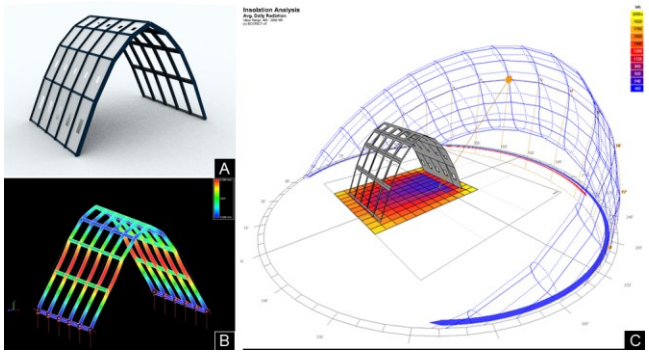

Figure 7. (A) Hand coded solution. (B) FEA structural deformation analysis. (C) Solar analysis.

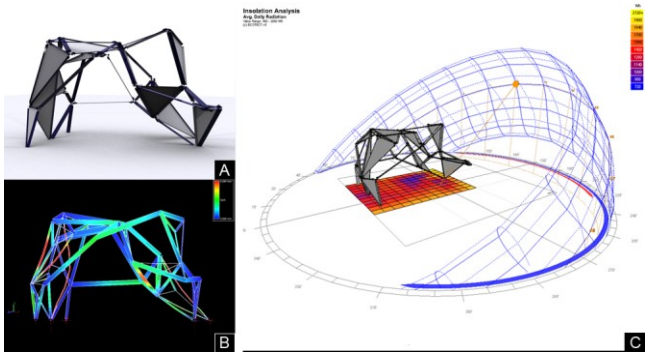

Figure 8. (A) Typical evolved solution. (B) FEA structural deformation analysis. (C) Solar analysis.

Figure 9 presents a comparative analysis of the hand coded structure against the (average) fitness of the best performing evolved structures over 10 runs. The hand coded performance breaks down as follows: structural fitness = 1, solar fitness = 0.75, height fitness = 0.58, cost fitness = 0.49 and overall fitness = 0.77. In comparison the (best) evolved structures over 10 runs have the following (average) performance: structural fitness = 0.92, solar fitness = 0.61, height fitness = 0.43, cost fitness = 1 and overall fitness = 0.72.

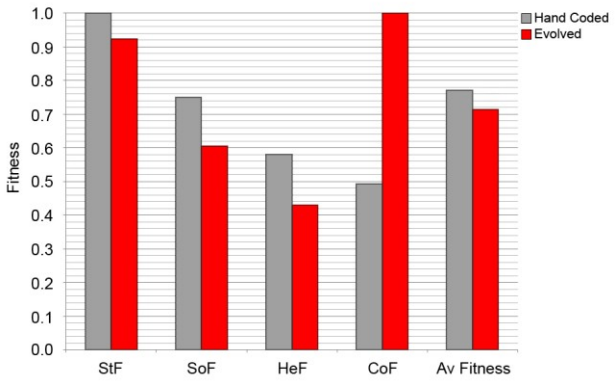

Figure 9. Comparative fitness analysis

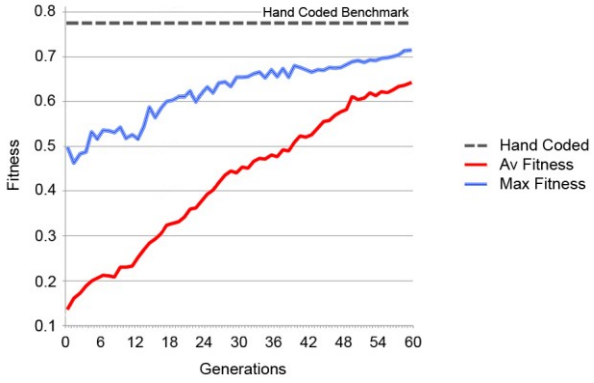

Figure 10. Average and maximum fitness over 10 runs

Figure 10 illustrates how the results of the GA converge in comparison to the hand coded fitness benchmark. Although we note that the average fitness is still rising when the runs were terminated, this was due to the time required for evaluation of each generation (the simulations are extremely processor-intensive). Future work will extend the number of generations to further investigate convergence properties, and assess whether the structural benefits obtained from longer runs are outweighed by the additional computational resources required.

4.5 Analysis

Figure 9 demonstrates that our evolved structures exhibit competitive fitness in comparison to the hand coded solution. After 60 generations the evolved structures obtain fitness only 6.5% below the hand-coded benchmark (Fig 10). Significantly they achieve this *without* prior knowledge of the solution or direct human guidance.

In this experiment, initial structures generated by random search provide extremely poor structural performance (Fig 11a-b). Consequently our fitness function was used to reward structural integrity to facilitate the evolution of build-able structures (Fig 9). Figure 10 shows that the maximum fitness derived from a random search is relatively high, suggesting that an exhaustive search may produce well performing structures. However, this is somewhat deceptive and highlights the challenge of designing a fitness function which can effectively communicate our *design intent* for real-world structures. Figure 11c-f illustrates structures derived by random search that exhibit relatively high fitness values (around 0.5). These structures do not provide well "rounded" architectural performance. However, they do exhibit good *structural performance* and *material economy,* which allows them to obtain deceptively high overall fitness values.

For the purpose of addressing this simple performance-based design problem, our fitness function facilitates the synthesis of shape, topology and material distribution to produce novel architectural solutions (Fig 12). However, we argue that non-trivial design problems will require *implicit* methods of describing *design intent,* so that real-world characteristics, such as structural integrity and material economy, become *emergent* properties that do not require *explicit* definition within the fitness function. We consider these in more detail in the following discussion.

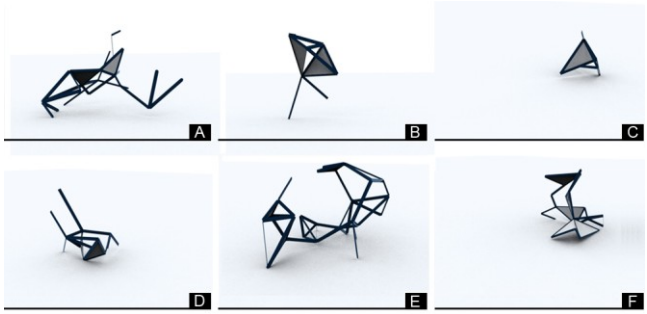

Figure 11. Structures generated by random search.

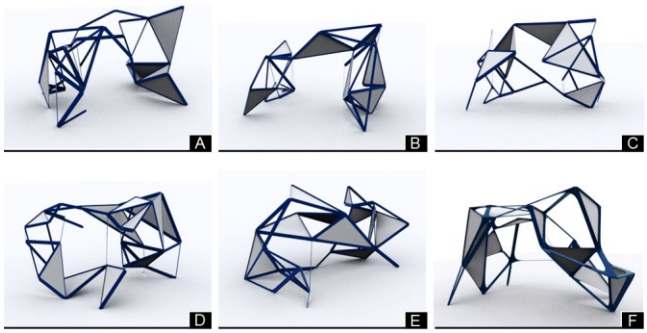

Figure 12. Examples of evolved structures

4.5.1 *Evolved Formal Traits*

During our experiments, evolved structures exhibited three *emergent*, recurring *formal traits*. Firstly, we observed the evolution of primitive *arch formations* (Fig 13a). Arch-shaped structures were assigned higher height fitness (HeF) due to the method of analysis, and were often *cheaper* to fabricate, since they eliminated unnecessary components from the middle of the three-dimensional volume. Secondly, we observed the evolution of *north-south differentiation* in structures. The north sides of structures often evolve to be structurally robust (Fig 13a left) which allows the south side of structures (Fig 13 right) to become structurally dependent, effectively "leaning up" against the robust north side. This formal trait was beneficial to structural performance, but also allowed the south sides of structures to express more numerous "petal-like" surface components in order to enhance solar shading. Thirdly, we observed *dual functionality* of surface components, to both enhance solar performance and improve structural integrity. Figure 13b-d illustrates the structural stresses imposed upon a typical arch structure. On the south (right) side of the structure, surface components are efficiently supported to increase shading. However, we also see surface components being utilized in areas of high structural stress to enhance *structural integrity*. This can be seen in Figure 13d on the north (left) side of the structure.

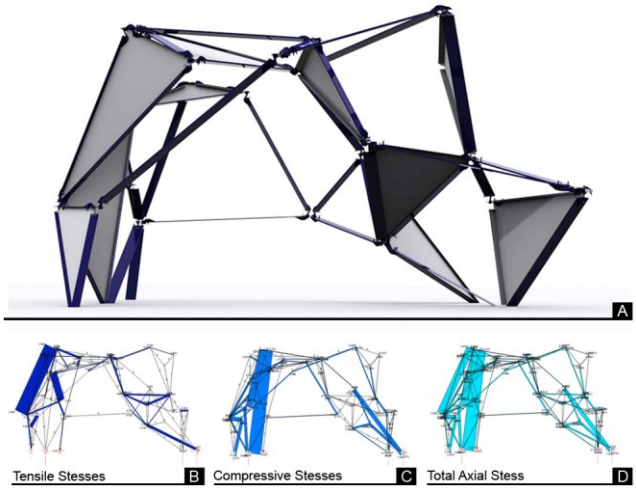

Figure 13: (A) Emergent arch formation. (B) Tensile stresses. (C) Compressive stresses. (D) Total axial stresses.

5. SUMMARY AND FUTURE WORK

We have presented a developmental genotype-phenotype mapping process and used it to self-organize a programmable library of architectural components in response to real-world performance objectives. Our results demonstrate a proof-of-principle that our evo-devo model can automatically synthesize shape, topology and material distribution, *in silico*, to produce functional architectural structures that are comparable to simple human designed solutions. This method extends the field of computational design synthesis and suggests that an evo-devo approach to architectural design may provide a way of breaking through the "glass ceiling" imposed by existing semi-automated design processes.

We claim this ability to automatically synthesize performance-driven architectural form via self-organizing processes is novel, and could open up entirely new territories for *sustainable design* of complex architectural structures.

Four key areas require further research before we can successfully apply *programmable self-organization* to non-trivial, physical architectural design problems.

Firstly, the current implementation of our model is dependent upon commercial CAD software packages to run performance analysis. The primary benefit of using existing CAD packages is (1) the validity of results and (2) they provide a useful medium for interdisciplinary communication. However, the central drawback is dramatically increased computation time. Each run of our model takes around 72 hours to complete. Consequently, the results presented have been limited to 60 generations and simply capture the beginning of evolutionary development. Future work requires the development of faster computational processes for the structural and environmental analysis of three-dimensional structures.

Secondly, the developmental representation provides a significant reduction in genotype size by reusing information during the regulatory developmental embryogeny. However, there is potential to further reduce genotype sizing and improve scalability of the model by utilizing *grammatical encodings* to generate node values, instead of relying upon fixed length string genomes.

Thirdly, the principal advantage of our developmental representation is the ability to integrate the physical properties of architectural components and evolve real-world performance for three-dimensional structures. However, evolved structures are currently analyzed as fully assembled morphologies that, in reality, would require additional scaffolding during construction. Further work is required to consider how physical incremental assembly of architectural components can be managed, and ultimately automated.

Finally, future work in this domain will explore how localized perturbations of nodes, in response to structural deflection analysis, can be introduced into the developmental embryogeny so that fundamental real-world properties (such as structural integrity) can be completely removed from our fitness function. We suggest that the ability to *implicitly* select structures which exhibit *essential* performance attributes will be critical for addressing non-trivial design problems in future works. However, we also believe that *implicit selection* via localized perturbations might allow additional useful properties, such as structural robustness (in response to damage), to emerge "for free". Such developmental representations that allow fundamental physical properties to emerge "for free" could facilitate the production of increasingly efficient, multifunctional and sustainable architectural morphologies using an automatic process of programmable self-organization.

6. ACKNOWLEDGMENTS

Daniel Richards is supported by a Ph.D. studentship from the Manchester Institute for Research and Innovation in Art and Design (MIRIAD).

7. REFERENCES

[1] Hensel, M. Menges, A. Weinstock, M. 2010. *Emergent Technologies and Design: Towards a Biological Paradigm for Architecture*, Routledge: Oxon.

[2] Oxman, N. 2010. *Material-based Design Computation*. Doctoral Thesis, MIT.

[3] Menges, A. Ahlquist, S. 2011. *Computational Design Thinking*. Sussex: Wiley.

[4] Oxman, R.: 2010. New Structuralism: Design, Engineering and Architectural Technologies. *Architectural Design*, 80(4), 14-23.

[5] Fleischmann, M. Lienhard, J. Menges, A. 2011. Computational Design Synthesis: Embedding Material Behaviour in Generative Computational Processes. In *Proceedings for eCAADe 2011*, (Ljubljana, Slovenia, September 21-24). 759-767.

[6] Hornby, G. 2004. Functional Scalability through Generative Representations: The Evolution of Table Designs. *Environment and Planning B*, vol 3. 569-587.

[7] Bentley, P. J. 2007. Climbing through Complexity Ceilings. In Burke, A. Tierney, T.(eds). *Network Practices: New Strategies in Architecture and Design*. New York: Princeton University Press. 178-197.

[8] Menges, A.: 2007. Computational Morphogenesis. In *Proceedings for ASCAAD 2007*, (Egypt, November 28-30). 725-744.

[9] Roudavski, S. 2009. Towards Morphogenesis in Architecture. *International Journal of Architectural Computing*, 7(3).345-374.

[10] Ulieru, M. Doursat, R. 2011. Emergent engineering: A radical paradigm shift. *International Journal of Autonomous and Adaptive Communications Systems,* 4(1). 39-60

[11] Kicinger, R. Arciszewski, T. and Jong, K. 2005. Evolutionary Computation and Structural Design: a Survey of the State of the Art, *Computers and Structures*, Vol 83. 1943-1978.

[12] Cladas, L. 2008. Generation of energy efficient architecture solutions applying GENE_ARCH: An evolution-based generative design system, *Advanced Engineering Informatics*, 22(1). 59-70

[13] Turrin, M. Buelow, P.V. Stouffs, R. and Kilian, A. 2010, Performance-Orientated Design of Large Passive Solar Roofs, In *Proceedings for eCAADe 2010*, Zurich, 321-330.

[14] Janssen, P. Frazer, J.H. Ming-Xi, T. 2002, Evolutionary Design Systems and Generative Design Processes, *Applied Intelligence*, 16(2). 119-128.

[15] O'Reilly, U. M. Hemberg, M. 2007. Integrating Generative Growth and Evolutionary Computation for Form Exploration. *Genetic Programming and Evolvable Machines*, 8(2). 163-186.

[16] Clune, J. Lipson, H. 2011. Evolving Three-Dimensional Objects with a Generative Encoding Inspired by Developmental Biology, In *Proceedings for ECAL 2011*, Paris, 141-148.

[17] Shea, K. Aish, R. Gourtovai, M. 2005. Towards Integrated Performance-Driven Generative Design Tools. *Automation in Construction*, vol 14. 253-264.

[18] Funes, P. and Pollack, J. 1998. Evolutionary Body Building: Adaptive Physical Designs for Robots, *Artificial Life*, Vol 4(4). 337-357.

[19] O.Stanley, K. Miikkulainen, R. 2003. A Taxonomy of Artificial Embryogeny, *Artificial Life*, Vol 9(2). 93-130.

[20] Bentley, P.J. And Kumar, S. 1999. Three Ways to Grow Designs: A Comparison of Embryogenies for an Evolutionary Design Problem, In *Proceedings for GECCO '99*, Florida. 35-43.

[21] Bentley, P. Kumar, S. (eds). 2003. *On Growth, Form and Computers*. London: Elsevier.

[22] von Mammel, S. Jacob, C. 2008. Evolutionary Swarm Design of Architectural Idea Models. *In Proceedings for GECCO '08*, (Atlanta, July 12-16). 143-150.

[23] O'Neill, M. McDermott, J. Swafford, J. M. Byrne, J. Hemberg, E. Brabazon, A. Shotton, E. McNally, C. Hemberg, M. 2010. Evolutionary Design using Grammatical Evolution and Shape Grammars: Designing a Shelter. *International Journal of Design Engineering*, 3(1). 4-24.

[24] Rieffel, J. Valero-Cuevas, F. Lipson, H. 2009. Automated Discovery and Optimisation of Large Irregular Tensegrity Structures, *Computers and Structures*, Vol 87. 368-379.

[25] Sims, K. 1994. Evolving 3D Morphology and Behavior by Competition, *Artificial Life*, Vol 1(4). 353-372.

[26] Lipson, H. and Pollack, J. 2000. Automatic Design and Manufacture of Robotic Lifeforms, *Nature*, Vol 406. 974-978.

[27] Eggenberger, P. 1997. Evolving Morphologies of Simulated 3D Organisms based on Differential Gene Expression, *CEC03*, Vol 1. 191-198.

[28] Bongard, J. and Pfeifer, R. 2003. Evolving Complete Agents using Artificial Ontogeny, In Hara, F. and Pfeifer, R (Ed), *Morpho-Functional Machines: The New Species (Designing Embodied Intelligence),* Springer Verlag: Berlin. 237-258.

[29] Miller, J. 2004. Evolving a Self-repairing, Self-regulating, French Flag Organism. In *Proceedings for GECCO '04*, (Washington, June 26-30). 129-139.

[30] Hiller, J.D. Lipson, H. 2010. Evolving Amorphous Robots. In *Proceedings for Alife XII*, (Odense, August 19-23). 717-724.

[31] Yogev, O. Shapiro, A. A. Antonsson, E. K. 2010. Computational Evolutionary Embryogeny. *IEEE Transactions on Evolutionary Computation*, 14(2). 301-325.

[32] Watson, J. D. Baker, T. A. Bell, S. P. Gann, A. Levine, M. Losick, R. 2007. *Molecular Biology of the Gene*. New York: Benjamin Cummings.

[33] Raff, M.C. 1992. Social controls on cell survival and cell death. *Nature*, Vol 356. 397-400.