18

Design of Self-Assembling, Self-Repairing 3D Irregular Cellular Automata

David Huw Jones[1], Richard McWilliam[2] and Alan Purvis[3]
[1]*Imperial College London*
[2,3]*University of Durham*
England

1. Introduction

> The core idea is that nature, imaginative by necessity, has already solved many of the problems we are grappling with. Animals, plants, and microbes are the consummate engineers. They have found what works, what is appropriate, and most important, what lasts here on Earth. This is the real news of biomimicry: After 3.8 billion years of research and development, failures are fossils, and what surrounds us is the secret to survival (Benyus, 1998).

Morphogenesis is a distributed chemical process that is responsible for co-ordinating the self-assembly and self-repair of biological systems. One example of morphogenesis in action is the ability for the human liver to sustain damage to over 75% of its mass and still repair itself.

Another demonstration of the reliability of morphogenesis is evident in the salamander family. Salamanders can regrow the same limbs repeatedly, as well as their tail, jaw, and the lenses and retinas of their eyes. In terms of body mass alone, the salamander can regenerate approximately 60% of itself in the event that it is damaged.

A final, remarkable, demonstration is the self-repair capability of the ascidian (a type of marine filter feeder). They have been reported to regenerate from just partial blood cells to give rise to a fully functional organism (Berrill & Cohen, 1936).

The aim of this research is to apply this robust self-assembly strategy to the design of self-assembling robotics. Here we describe various models for morphogenesis and existing techniques for designing self-assembling robotics. Then we introduce our cellular automata model for morphogenesis and determine the necessary conditions for its robust self-assembly and self-assembly to a pre-defined shape. Finally we demonstrate it co-ordinating the self-assembly of a 55,000 cell virtual robot.

2. Morphogenesis

Imagine an island of cannibals and missionaries. The cannibals can have sex, creating other cannibals in the process. The missionaries cannot have sex but own bicycles; two missionaries convert a cannibal into another missionary. This is the analogy Alan Turing (Turing, 1950) used to describe his "Reaction-Diffusion" model of morphogenesis. On such an island pockets of cannibals surrounded by missionaries are formed (see figure 1).

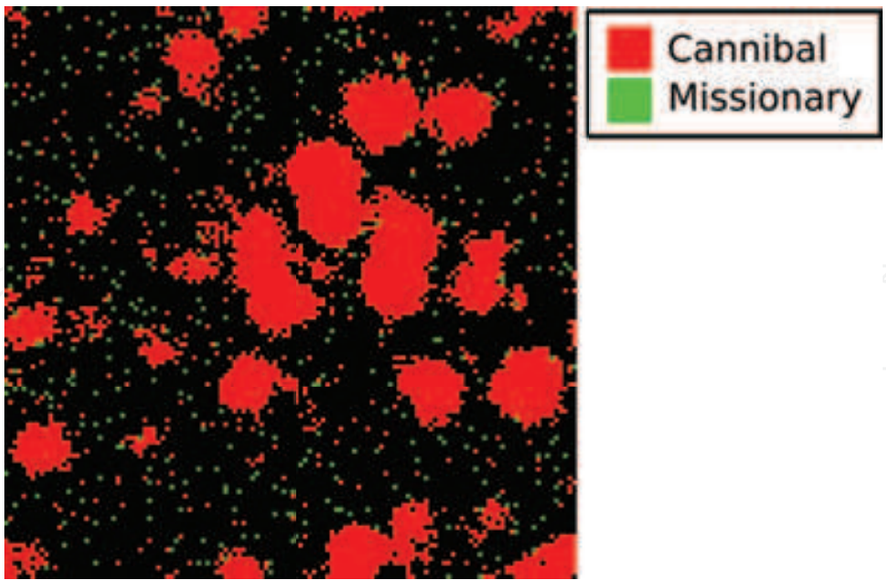


Fig. 1. Clumping of cannibals and missionaries formed by Turing's reaction-diffusion analogy

Turing proposed that the spots of a cheetah, the stripes of a zebra and every other arrangement of cells within living systems could be explained by the diffusion of chemicals he named "morphogens" and their interaction with each other.
To demonstrate this model, let us consider a simple two-morphogen (x_1, x_2) system of cells in one-dimension. [1]

2.1 The resting state

If, for a moment, we consider small perturbations of the concentrations of the morphogens about $\bar{x_e}$, the system's resting state, we can use linear ordinary differential equations (ODEs) (1) to describe their reaction and diffusion.

$$\frac{d\bar{x}}{dt} = \mathbf{A}\bar{x} \tag{1}$$

Where $\mathbf{A} = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix}$.
Let λ be an eigenvalue of $\mathbf{A}$. According to Cramer's rule, the system will only have non-trivial solutions if $det(\mathbf{A} - \lambda I) = 0$.

$$det(\mathbf{A} - \lambda I) = a_{11}a_{22} - \lambda(a_{11} + a_{22}) + \lambda^2 - a_{12}a_{21} \tag{2}$$

Let

$$T = a_{11} + a_{22} \tag{3}$$

$$D = a_{11}a_{22} - a_{12}a_{21} \tag{4}$$

[1] Proof derived from Turing's paper (Turing, 1950) and notes from Blowey (Blowey, 2007) and Childress (Childress, 2005).

Therefore $det(\mathbf{A} - \lambda I) = \lambda^2 - T\lambda + D$, its roots are given by:

$$\lambda = \frac{T \pm \sqrt{T^2 - 4D}}{2} \tag{5}$$

For the resting state to be stable (change from $\bar{x_e}$ is opposed), both roots must be negative. Therefore $D \in [0, \frac{T^2}{4}]$ and $T < 0$.

2.2 The transient state

Now let us consider larger permutations of the concentrations of the morphogens, as a result of diffusion of the morphogens between neighbouring cells.
Let each cell be a small cube of side $\triangle$ with the chemicals $\bar{x}$ distributed equally within it. Each cell has an index i from 0 to N and the cells are arranged in a 1 loop such that the neighbours of cell x^N are x^{N-1} and x^0.
Ficke's law states:

> The flow of chemical from one cell to another is proportional to the difference in the chemical concentrations of the two cells, the flow being from the higher to the lower.

The flux, f_j, of morphogen j into cell i, x_j^i, will be:

$$\text{flux } f_1 = k_1 \triangle^2 (x_1^{i-1} - x_1^i) + k_1 \triangle^2 ((x_1^{i+1} - x_1^i) \tag{6}$$

$$= k_1 \triangle^2 (x_1^{i-1} - 2x_1^i + x_1^{i+1}) \tag{7}$$

$$f_2 = k_2 \triangle^2 (x_2^{i-1} - 2x_2^i + x_2^{i+1}) \tag{8}$$

where $\{k_1, k_2\} > 0$ are constants of proportionality. Now

$$\frac{dx^i}{dt} = f(\bar{x}^i) + \mathbf{M}(\bar{x}^{i-1} - 2\bar{x}^i + \bar{x}^{i+1}) \tag{9}$$

where $\mathbf{M} = \begin{bmatrix} \triangle^2 k_1 & 0 \\ 0 & \triangle^2 k_2 \end{bmatrix}$ and $f(\bar{x}^i)$ determines the proportion of $\bar{x}^i$ that remains in cell i.
Note that because the cells form a loop, the system is closed and $\sum_i x^i$ is constant.
If the width, $\triangle$, of each cube tends to 0 we can consider the system continuous. If $s = \triangle .i$ then:

$$\frac{dx(s)}{dt} = f(s) = \mathbf{M}(\bar{x}(s - \triangle) - 2\bar{x}(s) + \bar{x}(s + \triangle)) \tag{10}$$

If we expand each term into its Taylor series through terms in $\triangle^2$ then:

$$\bar{x}(s + \triangle) = \bar{x}(s) + \frac{d\bar{x}(s)}{ds} \triangle + \frac{d^2\bar{x}(s)}{ds^2} \frac{\triangle^2}{2} + \cdots \tag{11}$$

$$\therefore f(s) \simeq \mathbf{M} \triangle^2 \frac{d^2\bar{x}(s)}{ds^2} \tag{12}$$

Let us assume $\triangle \to 0$, the number of cells, $N \to \infty$, $N \triangle \to L$, the circumference of the ring. We want to study the linear stability of the resulting continuous partial differential equation (PDE) in time.

$$\frac{\partial \bar{x}(t,s)}{\partial t} = \mathbf{A}\bar{x}(t,s) + \mathbf{M}\frac{\partial^2 x}{\partial s^2}(s,t) \tag{13}$$

Where $\mathbf{M}$ has been redefined as $\begin{bmatrix} \mu_1 & 0 \\ 0 & \mu_2 \end{bmatrix}$, $\mu_i = k_i \frac{\triangle^4}{2}$. μ_1, μ_2, the diffusion coefficients are finite and positive.
As the cells are in a loop we can look for pattern waves of the form:

$$x = e^{\sigma t + jks} x_0 \tag{14}$$

If for some μ_1, μ_2, k there exists a solution of this form such that $\Re(\sigma)$ is positive, the concentrations of the morphogens within the tissue will form stable patterns.
Using the identity $\frac{d^2}{ds^2} e^{jks} = -k^2 e^{jks}$ and the product rule, the $\frac{\partial^2 \bar{x}}{\partial s^2}$ component of (14) can be found:

$$\frac{d^2 \bar{x}}{ds^2} = -k^2 \bar{x} \tag{15}$$

Substituting (15) into (13) gives:

$$\frac{d\bar{x}}{dt} = \mathbf{A}\bar{x} + M \begin{bmatrix} -k^2 & 0 \\ 0 & -k^2 \end{bmatrix} \bar{x} \tag{16}$$

$$= \left(\mathbf{A} + \begin{bmatrix} \mu_1 k^2 & 0 \\ 0 & \mu_2 k^2 \end{bmatrix} \right) \bar{x} \tag{17}$$

Thus if $\mathbf{B} = \mathbf{A} - \begin{bmatrix} \mu_1 k^2 + \sigma & 0 \\ 0 & \mu_2 k^2 + \sigma \end{bmatrix}$ the pattern wave solutions are determined by the non-trivial solution to $\mathbf{B}\bar{x_0} = 0$.
Again using Cramer's rule, the determinant of $\mathbf{B}$ must equal 0. Thus:

$$\sigma^2 + \tau\sigma + \psi = 0 \tag{18}$$

where

$$\tau = k^2(\mu_1 + \mu_2) - (a_{11} + a_{22}) \tag{19}$$

$$= k^2(\mu_1 + \mu_2) - T \tag{20}$$

$$\psi = D - \mu_1 k^2 a_{22} - \mu_2 k^2 a_{11} + \mu_1 \mu_2 k^4 \tag{21}$$

Both roots must have negative real components. Thus $\tau < 0$ and $\psi > 0$. For $D < 0$ both $a_{11}a_{22} < 0$ and $a_{12}a_{21} < 0$. For $\psi < 0$,

$$a_{22}\mu_1 + a_{11}\mu_2 > 0 \tag{22}$$

If $a_{11} < 0$ its corresponding entry in the matrix $\mathbf{B}$, $a_{11} - \mu_1 k^2 + \sigma$ is also negative. Thus the chemical, x_1 will decay to the rest state $\bar{x_e}$ in the absence of x_2. Turing called x_1 the inhibitor chemical and x_2 the activator chemical.

2.3 Turing instability inequalities

From (22) and (3) one can deduce that, for the system to form stable patterns $\mu_2 < \mu_1$. Thus the inhibitor chemical must diffuse faster than the activator chemical. Referring back to the "cannibals and missionaries" analogy, the presence of two missionaries in any given square inhibits the progress of the diffusion of the cannibals across the island, however because the inhibitor-missionaries own bicycles they diffuse more rapidly than the activator-cannibals.
Equation (21) is a quadratic in k^2, the minimum value of which is:

$$\psi_{min} = D - \frac{(a_{22}\mu_1 + a_{11}\mu_2)^2}{4\mu_1\mu_2} \tag{23}$$

Therefore:

$$a_{22}\mu_1 + a_{11}\mu_2 > 2\sqrt{\mu_1\mu_2 D} \tag{24}$$

2.4 An example reaction-diffusion system

An example two-morphogen system that meets (24), (22), (3) and (4) has the following reaction equations:

$$x_1(\bar{x}) = (16 - x_1x_2)s \tag{25}$$

$$x_2(\bar{x}) = (x_1x_2 - x_2 - 20)s \tag{26}$$

To find the unique equilibrium we set (25) and (26) to zero. Thus $x_{e1} = -4$ and $x_{e2} = 4$.
Assuming the system is linear about $\bar{x_e}$ we can differentiate (25) and (26) to form $\mathbf{A}$.

$$\mathbf{A} = \begin{bmatrix} -x_2 s & -x_1 s \\ x_2 s & x_1 s - s \end{bmatrix} = s \begin{bmatrix} -4 & 4 \\ 4 & -5 \end{bmatrix} \tag{27}$$

Figure 2 shows the results of this model.

2.5 Experimental evidence for morphogenesis

A two-morphogen system exists in a developing human body, shortly after the creation of the blastula. Two proteins, the BCD protein and the HB-M protein, create a localised determinant centered about the zygote (Griffiths, 1976). Since these proteins have different diffusion rates, and interact with each other, spatial concentration patterns form. Different concentrations activate different genes in the genome of each cell. These gene selections in turn correspond to different types of cell. Thus the beginnings of form and cellular differentiation appear in the developing embryo.
A three-morphogen system exists in the developing fruit fly. The morphogens, bicoid, eve and caudal, are important for patterning the head, thorax and abdominal regions of the embryo (Rosee et al., 1997). Figure 3 shows the concentrations of each morphogen shortly after fertilisation. Figure 4 shows various other morphogenesis patterns that have been sampled during the development of the fruit fly.

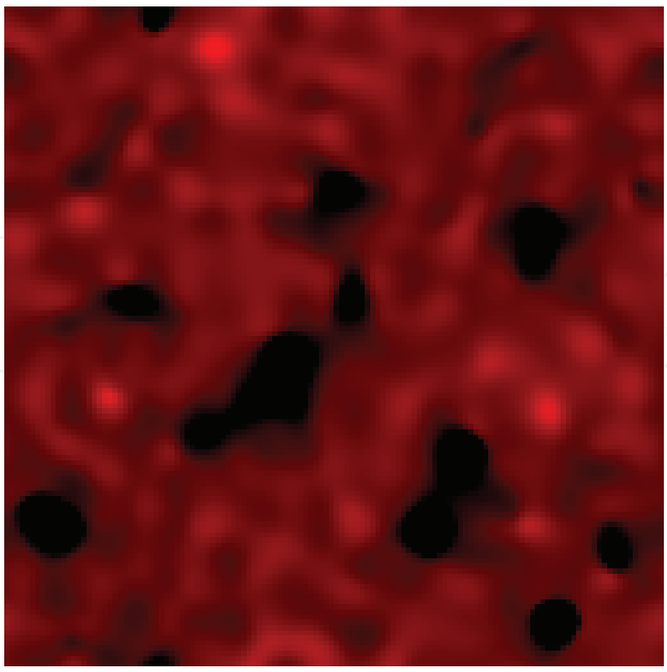


Fig. 2. Example of Turing's reaction-diffusion equations

2.6 A more comprehensive model of the developmental cycle

But experimental results suggests that control of the developmental cycle is not as simple as systems of interacting morphogens. Most tissues have asymmetric distributions of various properties: for example, in a hydra the nerve cell density is much greater in the head than the body. Transplantation experiments (Morgan, 1904) suggest that this polarity affects the decision of where to form structures relative to the tissue.

Further work by the experimental biologist John Gurdon (Gurdon, 1968) suggests that the concentration levels of morphogens within a tissue are typically very small and undetectable by the majority of cells. To ensure each cell in this tissue develops in line with the rest of its surrounding cells, each cell that detects the presence of a signalling protein transmits a chemical signal to its neighbouring cells. This is known as cell-to-cell communication, or the "community effect".

Meinhardt (Meinhardt, 1982) sought to incorporate these theories of development into a more comprehensive model than that proposed by Turing. The following model, one of many he proposed, uses five reaction-diffusion equations to form striped patterns.

$$\frac{\partial g_1}{\partial t} = \frac{c g_1^2}{r s_1} - \alpha g_1 + D_g \frac{\partial^2 g_1}{\partial x^2} + \rho_0 \tag{28}$$

$$\frac{\partial g_2}{\partial t} = \frac{c g_2^2}{r s_2} - \alpha g_2 + D_g \frac{\partial^2 g_2}{\partial x^2} + \rho_0 \tag{29}$$

$$\frac{\partial r}{\partial t} = \frac{c g_1^2}{s_1} + \frac{c g_2^2}{s_2} \tag{30}$$

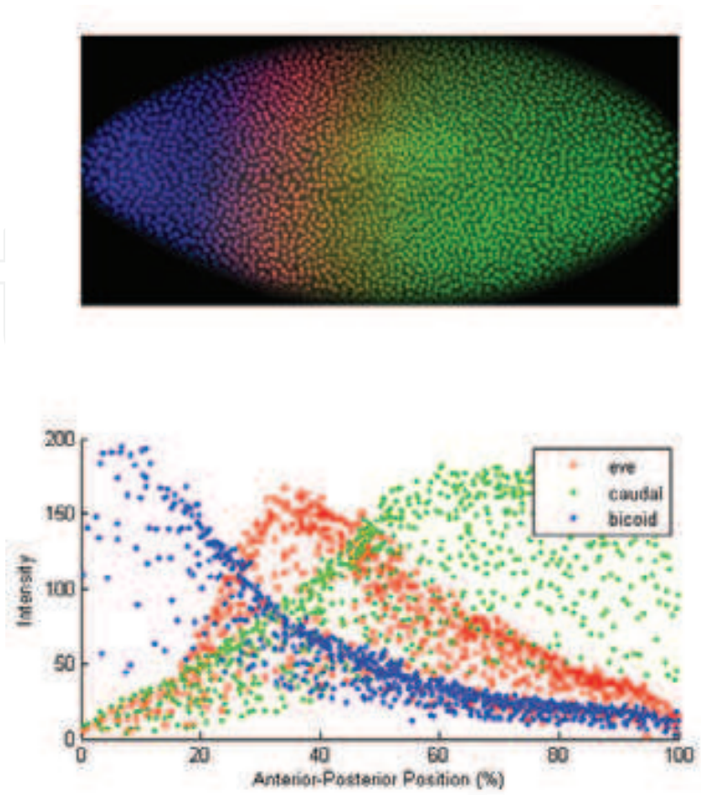


Fig. 3. Anterior-posterior determination in fruit flies. Image courtesy of the FlyEx database (Pisarev et al., 2008)

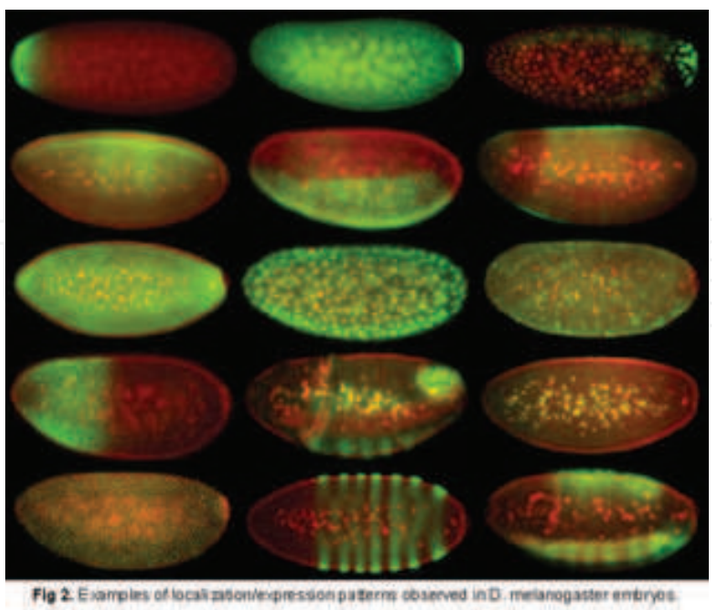


Fig. 4. Morphogenesis patterns in fruit flies
Image courtesy of Dr. Eric Lecuyer, Canadian Institute of Health Research

$$\frac{\partial s_1}{\partial t} = \gamma \frac{g_1 - s_1}{D_s} \frac{\partial^2 s_1}{\partial x^2} + \rho_l \tag{31}$$

$$\frac{\partial s_2}{\partial t} = \gamma \frac{g_2 - s_2}{D_s} \frac{\partial^2 s_2}{\partial x^2} + \rho_l \tag{32}$$

g_1 and g_2 are short-range activator substances. They form mutually exclusive feedback loops. Each is subject to long-range inhibitor substances, s_1 and s_2. That they are mutually exclusive is ensured by equation (30).
Figure 5 shows a result of the model with different diffusion rates D_g and D_s.

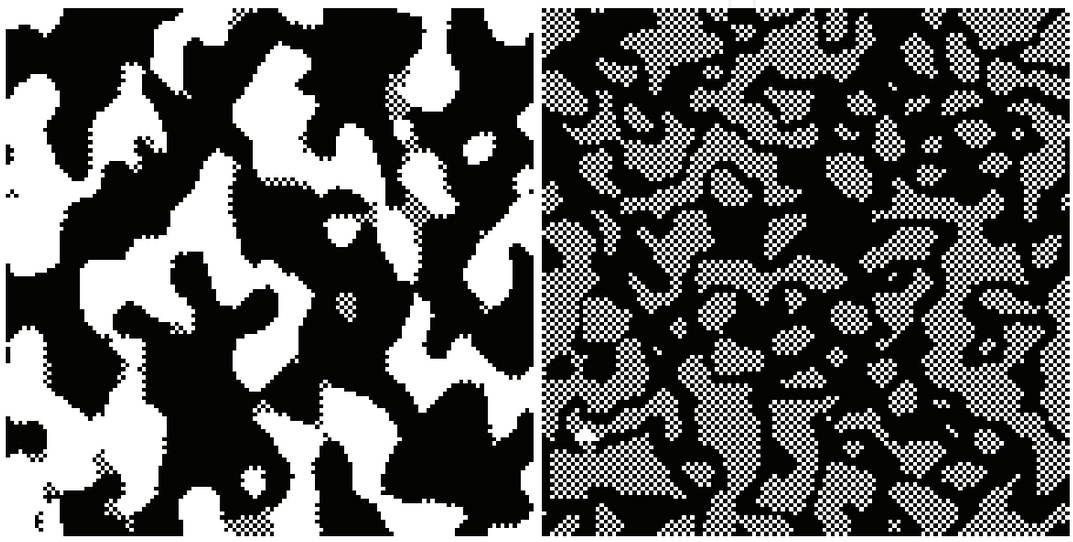


Fig. 5. Example of Meinhardt stripes

3. Morphogenesis for self-assembling robotics

Within the field of robotics, self-assembly describes the spontaneous aggregation of pre-formed cells into well-defined, stable assemblies. This assembly happens without the assistance of any external co-ordinating processes.
The design of large self-assembling systems has been the bailiwick of chaos mathematicians and computational evolutionists for the last ten years because the relationship between locally-interacting cells and the general form of the larger assembly is not well understood. It has, however, been identified as essential to the development of future robotics (Yim et al., 2007).
To complicate matters further, a common assumption is that each pre-formed cell is identical. This means that each cell can take the place of another such that the replacement of damaged cells is trivial. Also the challenge of self-replication is reduced from the procedure of copying the entire system once to that of copying a small cell of the system many times.
There are various existing techniques for the design of self-assembling systems:
Cartesian co-ordinate mapping. One possible way of organising self-assembly is with a co-ordinate system. Each cell works out where it is in the system from the co-ordinates of its neighbours. A map, stored in each cell, that relates each co-ordinate to a particular type of cell

is then used to determine what form each cell should take. This is the basis of the embryonics algorithm proposed by Tyrrell (Canham & Tyrrell, 2003).

Hierarchical maps. Murata (Murata et al., 1999) proposed a hierarchical map for the self-assembly of mechanical structures from "fractum", small motorized robots capable of connecting to one another. The assembly program, stored in each cell, contains a description of the neighbourhood of every cell within a simple structure. Cells move randomly until every cell has a correct neighbourhood, at which point every cell freezes. Then another simple structure starts to form about one of the frozen cell. Because repeated simple structures need only be coded once, this algorithm is more efficient than a cartesian co-ordinate mapping.

Ordinary differential equations (ODE). Fleischer (Fleischer & Barr, 1993) studied the effects of biological cell migration, cell hereditary and inter-cellular communications using an ODE model of locally-interacting cells. By using an adaptive euler-solver developed by Barzel (Barzel, 1992), Fleischer was able to design the model to self-assemble into interesting shapes. He concluded that it was difficult to design cells to assemble to specific forms, and went on to suggest the use of evolution as an alternative approach.

Unsupervised evolutionary algorithms. Eggenberger (Eggenberger, 1997) used a multi-cell model to study the effects of genetic lineage and inter-cellular chemical interactions on biological tissue development. A genetic algorithm was used to design the model to converge to interesting patterns. The fitness of each solution was tested against an arbitrary model size and the symmetry of the system about the x-axis.

Supervised evolutionary algorithms. Most recent experiments have relied on supervised evolutionary algorithms for the design of self-assembling systems. For instance, Izumi (Kizotaka et al., 1999) studied supervised genetic algorithms for the self-assembly of Murata's "fractum". Miller (Miller & Banzhaf, 2003) used a 2D array of 256 cells to simulate the inter-cellular chemical interactions of a biological tissue during development. A genetic algorithm was used to design the model to self-assemble to a single pattern (3 vertical stripes) from null or partially corrupt (up to 25%) initial conditions.

Deterministic algorithms for the design of convergent cellular automata (CA). Previous work by the authors (Jones et al., 2008), which this chapter will build upon, has been on the design of convergent cellular automata.

We are going to use a cellular automata model of morphogenesis in order to determine the necessary conditions for robust self-assembly.

4. The conditions for cellular automata convergence

Cellular automata (CA) are dynamic systems in which space and time are discrete. CA consist of a number of identical cells pre-arranged into an array. Each cell can be in one of a number of states. The next state of each cell is determined at discrete time intervals according to the current state of the cell, the current state of neighbouring cells and a next-state rule that is identical for each cell. Let us index each cell with the tuple (i, j), then describe the state of each cell with an integer, $c_{i,j,t}$ and the pattern of the entire array as a matrix, $\mathbf{C}_t$.

If $\mathbf{C}_0$ is the initial pattern of cell states, $f(\mathbf{C}_0)$ is its subsequent pattern after one time step, and $f(f(\mathbf{C}_0))$ or $f^2(\mathbf{C}_0)$ is its pattern at $t = 2$; where f() describes the transition from one iteration to the next. The matrix $\mathbf{C}_t$ is first transcribed into a row-major vector, $\overline{C_t}$ in order for $f()$ to be a linear function of matrix algebra.

Let us define a simple transition function for the next time step:

$$c_{i,j,t+1} = uc_{i,j-1,t} + lc_{i-1,j,t} + rc_{i+1,j,t} + ac_{i,j+1,t} + pc_{i,j,t} + d \tag{33}$$

Where u, l, r, a and p are coefficients of the state of neighbours of each cell, and of the state of the cell itself.

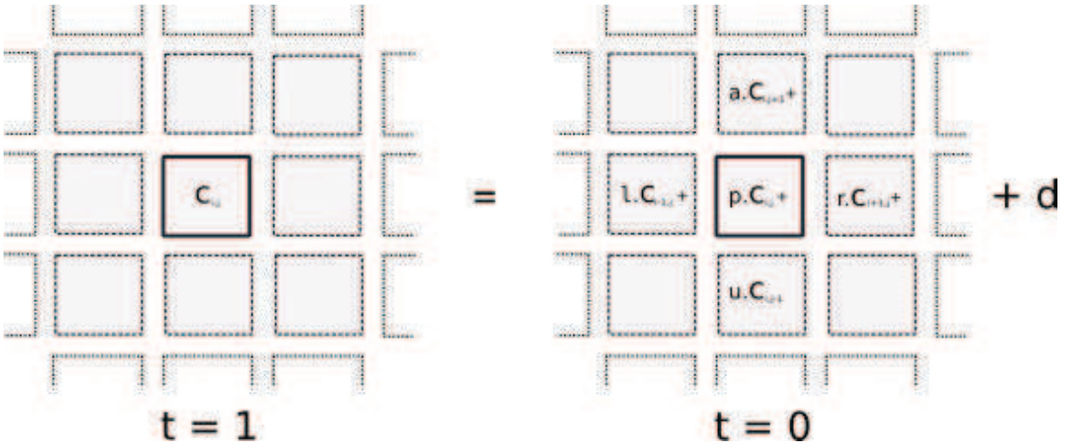


Fig. 6. A simple linear transition function

A transition function for the entire array can be formed from (33) such that $f(C_t) = \mathbf{A}\overline{C_t} + \overline{D}$ where $\overline{D}$ is a constant and the transition matrix (for a three by three CA), $\mathbf{A}$, takes the form:

$$\mathbf{A} = \begin{pmatrix} p & l & 0 & u & 0 & 0 & 0 & 0 & 0 \\ r & p & l & 0 & u & 0 & 0 & 0 & 0 \\ 0 & r & p & 0 & 0 & u & 0 & 0 & 0 \\ a & 0 & 0 & p & l & 0 & u & 0 & 0 \\ 0 & a & 0 & r & p & l & 0 & u & 0 \\ 0 & 0 & a & 0 & r & p & 0 & 0 & u \\ 0 & 0 & 0 & a & 0 & 0 & p & l & 0 \\ 0 & 0 & 0 & 0 & a & 0 & r & p & l \\ 0 & 0 & 0 & 0 & 0 & a & 0 & r & p \end{pmatrix} \tag{34}$$

The spacing of the coefficients a, r, p, l and u within $\mathbf{A}$ depend on the size of the CA.
By the repeated application of $f()$, the transition from $\mathbf{C}_0$ to $\mathbf{C}_n$ (where $n > 1$) becomes:

$$f^2(\overline{C_0}) = \mathbf{A}(\mathbf{A}\overline{C_0} + \overline{D}) + \overline{D} \tag{35}$$
$$f^3(\overline{C_0}) = \mathbf{A}(\mathbf{A}(\mathbf{A}\overline{C_0} + \overline{D}) + \overline{D}) + \overline{D} \tag{36}$$
$$f^3(\overline{C_0}) = \mathbf{A}^3\overline{C_0} + \mathbf{A}^2\overline{D} + \mathbf{A}\overline{D} + \overline{D} \tag{37}$$

This can be expanded to form:

$$f^n(\overline{C_0}) = \mathbf{A}^n\overline{C_0} + \mathbf{A}^{n-1}\overline{D} + \mathbf{A}^{n-2}\overline{D} + ... + \mathbf{A}\overline{D} + \overline{D} \tag{38}$$

Using the geometric series equation this can be simplified to form:

$$f^n(\overline{C_0}) = \mathbf{A}^n\overline{C_0} + (\frac{\mathbf{I} - \mathbf{A}^{n-1}}{\mathbf{I} - \mathbf{A}})\overline{D} \tag{39}$$

Equation (39) determines the pattern the CA forms after n iterations of the transition function (33) have been applied to every cell synchronously.

Given a sufficiently large n in order for the dynamic non-linear system to converge, the final pattern, $\overline{C_n}$, must be independent of the initial pattern, $\overline{C_0}$. Thus no matter what the starting pattern (where $t = 0$ refers to the initial pattern or any pattern that might be the result of system corruption), the pattern of cell states will always return to the same stable pattern.
Thus $\mathbf{A}^n$, the coefficient of $\overline{C_0}$, must equal zero. For this to be so, referring to the coefficients of the states of the cells above, below, left and right and of the cell itself respectively, the following must hold: either l or r must equal zero, either u or a must equal zero, p must equal zero. That is, $\mathbf{A}$ must be an upper-diagonal or lower-diagonal matrix.
Let us now analyse the conditions for convergence of a non-linear transition function, $g()$.

$$g(\overline{C_0}) = \sum_{i=0}^{k} \mathbf{A}_l \overline{C_0} \quad \text{where } A_i = \begin{pmatrix} p_i & l_i & 0 & u_i & 0 & \cdots \\ r_i & p_i & l_i & 0 & u_i & \\ 0 & r_i & p_i & 0 & 0 & \\ a_i & 0 & 0 & p_i & l_i & \\ 0 & a_i & 0 & r_i & p_i & \\ \vdots & & & & & \end{pmatrix} \tag{40}$$

If k is greater than three, expanding $g^n(C_0)$ gives a coefficient of C_0 formed by the multinomial of the transition matrices:

$$(\mathbf{A}_0 + \mathbf{A}_1 + \mathbf{A}_2 + \cdots + \mathbf{A}_m)^n \overline{C_0} \tag{41}$$

The multinomial expansion can be described as:

$$(x_1 + x_2 + \cdots + x_m)^n = \sum_{k_0,k_1\cdots,k_m} \binom{n}{k_0, k_1, \cdots, k_m} x_1^{k_0} x_2^{k_1} \cdots x_m^{k_m} \tag{42}$$

Where the summation is taken over all sequences of $k_1, k_2, \cdots, k_m$ such that:

$$\sum_{i=1}^{m} k_i = n \tag{43}$$

And the multinomial coefficient can be expressed as:

$$\binom{n}{k_0, k_1, \cdots, k_m} = \frac{n!}{k_0! k_1! \cdots k_m!} \tag{44}$$

Thus the coefficients of every $\overline{C_0}$ term are constructed from a multinomial coefficient and n members of the set of $\mathbf{A}$ transition matrices:

$$\mathbf{A} = \{\mathbf{A}_0, \mathbf{A}_1, \mathbf{A}_2, \cdots, \mathbf{A}_k\} \tag{45}$$

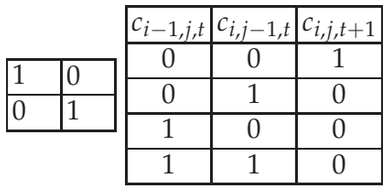
For every coefficient of $\overline{C_0}$ to be zero, every possible product of n members of $\mathbf{A}$ must be zero. Thus every member of $\mathbf{A}$ must be a lower-diagonal matrix or every member must be an upper-diagonal matrix.
As $g()$ is a sum-of-products function, it can be represented as a two-input one-output logic function. This logic function is best described as a list of rules that form the entries of a look-up table (LUT). Each unique combination of two input-values that is required by the design would form an entry in the LUT.

5. Determining the local rules for a regular CA to converge

Before we can consider the design of self-assembling systems, we need to examine the relationship between the next-state rule each cell obeys and the general form the system converges to. An algorithm is needed to design the next-state rule, $f()$, of a regular CA such that the automata converges to a specific pattern. A technique that works for a limited set of patterns simply assumes the automata has already reached this final pattern, derives the rules each cell must obey in order to stay in the same state then forces every other input combination to a next-state of zero. Figure 7(a) shows a simple example of a pattern to which we would like the automata should converge, figure 7(b) shows the next-state rule, $f()$, that each cell must obey such that the automata converges to this pattern. This example can also use an algebraic next-state rule:

$$c_{i,j,t+1} = 1 - c_{i-1.j.t} - c_{i,j-1,t} \tag{46}$$

Figure 8 shows the automata using equation (46) to converge from null (a) and random (b) initial conditions.



1	0
0	1

(a)

$c_{i-1,j,t}$	$c_{i,j-1,t}$	$c_{i,j,t+1}$
0	0	1
0	1	0
1	0	0
1	1	0

(b)

Fig. 7. A CA pattern and the necessary next-state rule for each cell

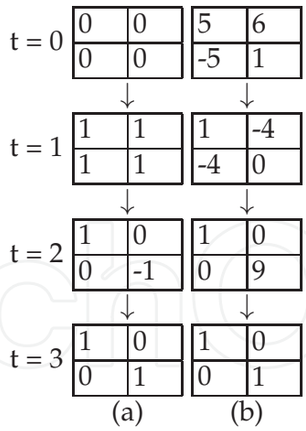


Fig. 8. Example CA convergence from null (a) and random (b) initial conditions

There still exist certain patterns that cannot be generated using this simple deterministic algorithm. This is because the same two input combinations cannot map to different output values. Figure 9(a) shows a pattern that cannot be formed using this approach if the inputs are from above and to the left of each cell.

This is because a portion of this pattern is impossible to form with $f()$ alone because it would require $f(2,2) = 2$ for cell $c_{2,3}$ and $f(2,2) = 1$ for cell $c_{3,3}$, something that is not permissible in a many-to-one mapping.

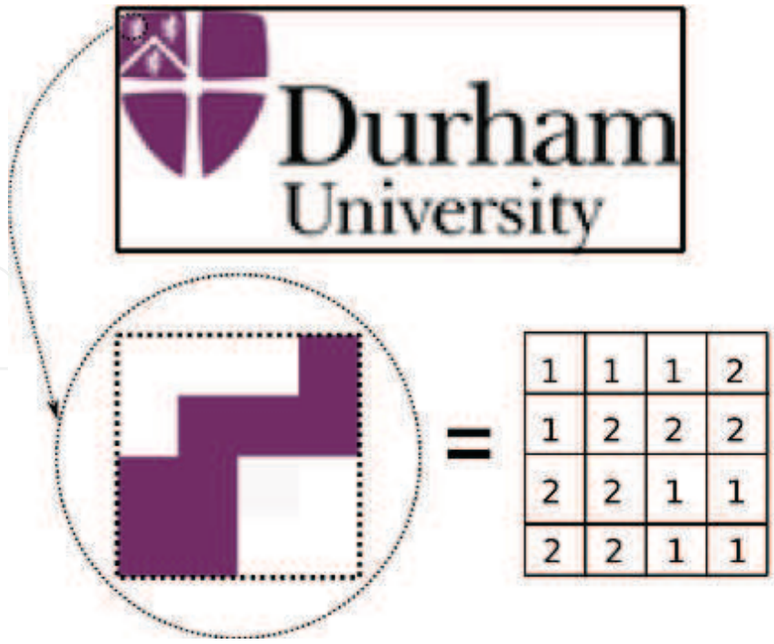


Fig. 9. A pattern for which the design is not trivial

To this end additional computation must be introduced in the form of a many-to-one state-to-output mapping function denoted by $g()$. Figure 10(a) shows an example implementation of how the next state and output of each cell might be calculated. The reader is referred to (Jones et al., 2008) for further information.

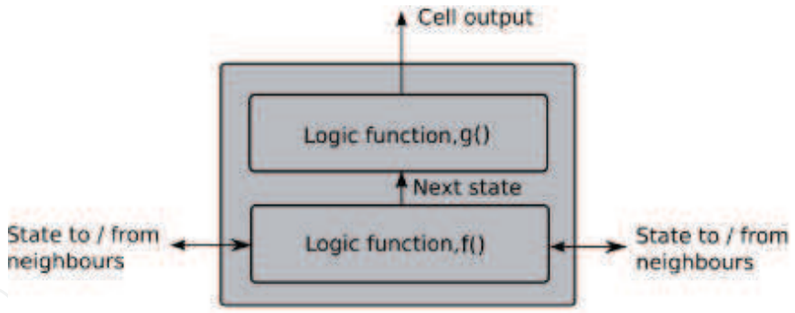
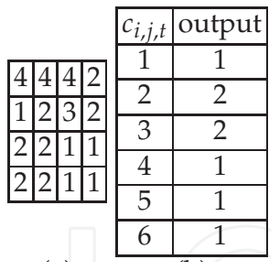


Fig. 10. A possible implementation of $f()$ and $g()$

Thus the pattern of figure 9(b) could be rewritten as figure 11(a). The state of cell $c_{1,2}$ has been replaced with the first available integer that does not form a mapping that contradicts previously determined mappings. Figure 11(b) is the mapping from figure 9(b) to figure 11(a) and figure 12 is the next-state rules each cell must obey.

By introducing as many state values for each output as is necessary, any desired pattern can be generated. However an optimum solution would use as few state values as possible. If the sequence in which cells are assigned a state value starts from the corner furthest from the direction of state-input, a minimum number of state values will be needed. Hence if the inputs are taken from above and to the left of each cell, the first cell to be assigned a value would be the cell in the bottom-right of the array.



4	4	4	2
1	2	3	2
2	2	1	1
2	2	1	1

(a)

$c_{i,j,t}$	output
1	1
2	2
3	2
4	1
5	1
6	1

(b)

Fig. 11. A solution to figure 9(b)

$c_{i-1,j,t}$	$c_{i,j-1,t}$	$c_{i,j,t+1}$
0	0	6
0	4	2
0	5	4
0	6	5
1	0	2
1	1	1
1	2	1
2	1	1
2	2	2
2	3	2
4	2	3
5	1	2
6	0	1

Fig. 12. The ruleset to figure 9(b)

Figure 13 shows the CA converging from null initial conditions. Figure 14 shows the CA converging from random initial conditions.
Implementing the design algorithm using a python script, the derivation for this 120 by 60 cell convergent CA took 180 seconds to complete on a 3.2GHz Intel Xeon processor.

6. Fundamental criteria for convergence of a self-assembling systems

We have so far only considered systems of cells which are pre-arranged into regular arrays. If the state of each cell also includes the position at which a cell should reside relative to other cells, the analysis can be extended to the design of self-assembling systems.
The scheme proposed so far relies on each cell computing its next-state from the current state of two neighbouring cells; the relative location of each neighbour to the cell is common to every cell in the array. Therefore if one cell updates according to the state of cells above and to the left of itself, so does every cell. In a rectangular array of cells, most will have two neighbours upon which they determine their next state from, some will have one neighbour and one will have no neighbours. In effect the desired pattern emerges from this one cell (the origin cell); as shown in figure 15(a). In the case of a partially-assembled CA, or a CA that converges to form an irregular shape, this is not necessarily the case. This is shown in figure 15(b).
A solution is to allow each cell to determine its next-state according to the current state of two neighbouring cells as before, but to determine which two cells (above or below, to the

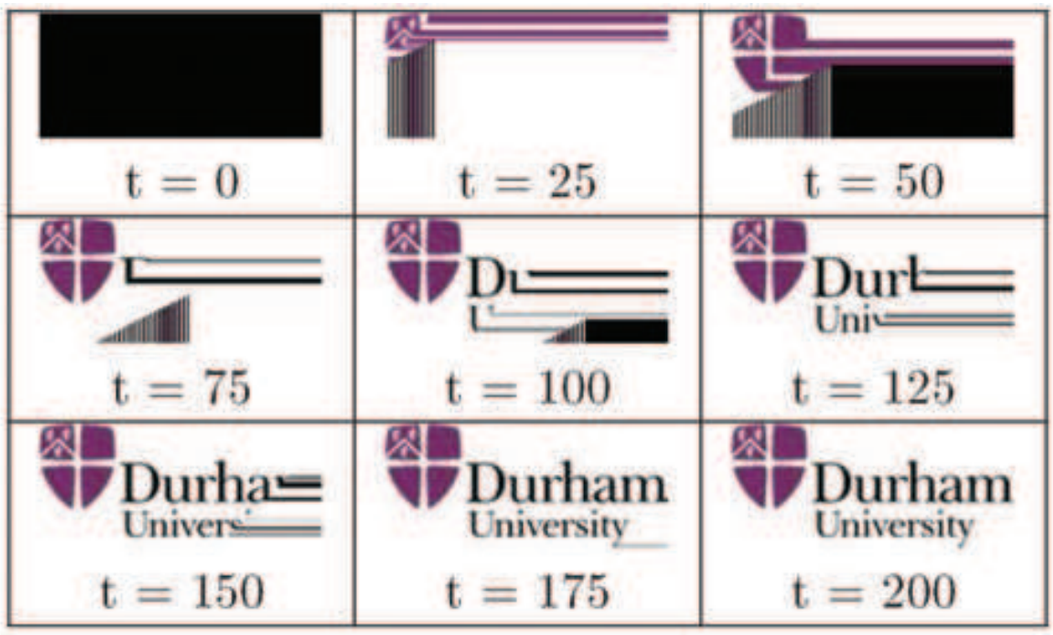


Fig. 13. The convergence of a 120 by 60 CA from null initial conditions

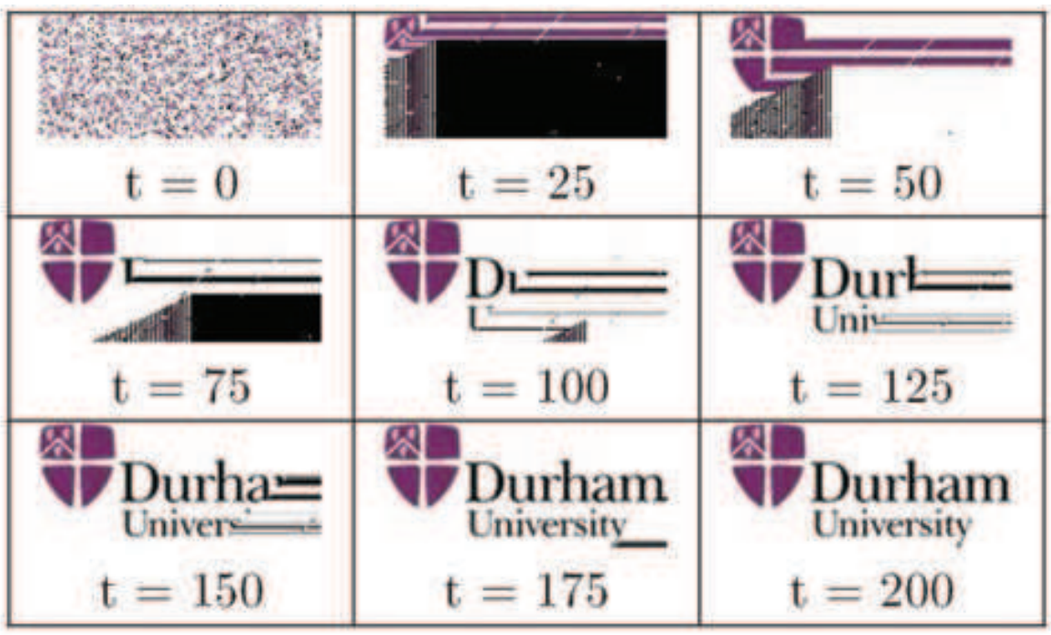


Fig. 14. The convergence of a 120 by 60 CA from corrupt initial conditions

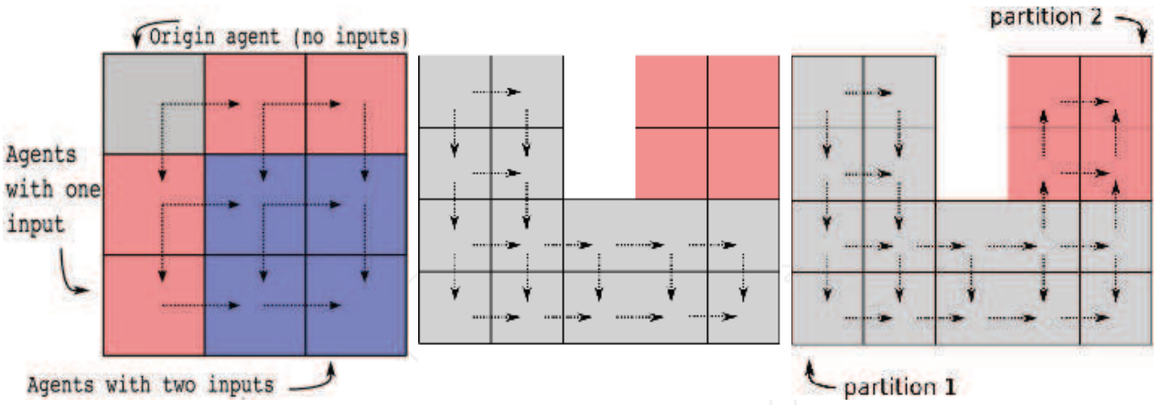


Fig. 15. Inputs combinations, an irregular 2D CA and the flow of state information from the origin cell

left or right) according to its current state. Thus, while one cell may determine its next-state according to cells above and to the left of itself, another might determine its next-state according to cells from below and to the right of itself. This "state to neighbourhood-function" is a many-to-one mapping (see figure 16).

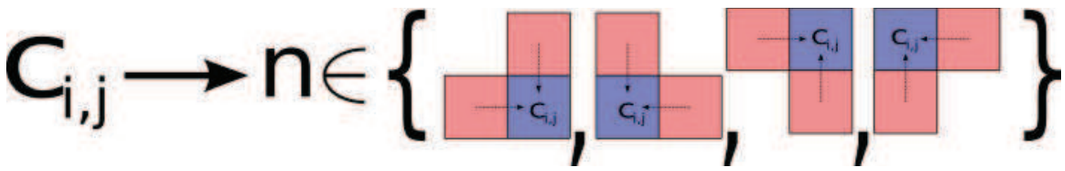


Fig. 16. The state-neighbourhood function mapping

If we divide the CA into partitions with common neighbourhood configurations (as shown in figure 15(c)), where each partition directly or indirectly derives its neighbouring states from the first partition ($P1$) which contains the origin cell, the pattern of this irregular CA effectively emerges from the origin cell. Thus provided $f()$ conforms to the requirements for the automata to converge, a multiple-partition automata with $f()$ as its transition function, will also converge to a steady pattern.

6.1 Determining the local rules for an irregular CA to converge to an arbitrary pattern

The design algorithm for a convergent irregular CA is similar to that for a regular CA. The desired pattern is first divided into partitions of cells with common neighbourhood configurations. This is done so as to maximise the size of each partition. This maximises the number of cells that have two neighbours in their neighbourhood configuration, thus reducing the number of state values needed for the design.

The design algorithm is then applied to each partition in turn. The sequence of partitions is reversed, so the first partition to be solved is the partition furthest from the origin cell. There are now three mappings against which each state-assignment must be tested:

1. The mapping from the current state of its neighbours to the next state of the cell ($g()$).
2. The mapping from the current state of the cell to its output ($h()$).
3. The mapping from the current state of the cell to the relative location of the neighbouring cells to which it transmits this state.

Every mapping created by each state assigned by the design algorithm must first be tested against each mapping previously created by the design algorithm, including those created for cells in different partitions.

6.2 The assembly algorithm

In previous demonstrations the automata has consisted of a pre-assembled array of cells. However the new algorithm makes the self-assembly of the automata possible. If un-used cells fasten themselves to used cells only where that cell is transmitting state-values, the automata will self-assemble from the origin cell until complete. This self-assembly depends on the origin cell already existing.

6.3 Demonstration

Figure 17 shows the five partitions of a 1000 cell 2D system. Figure 18 shows this system self-assembling and converging to the desired stable shape. Figure 19 shows this same system self-healing from a corrupted shape.
Implementing the algorithm of section 6.1 using a python script, the derivation for this 1000 cell self-assembling system took 30 seconds on a 3.2GHz Intel Xeon processor.

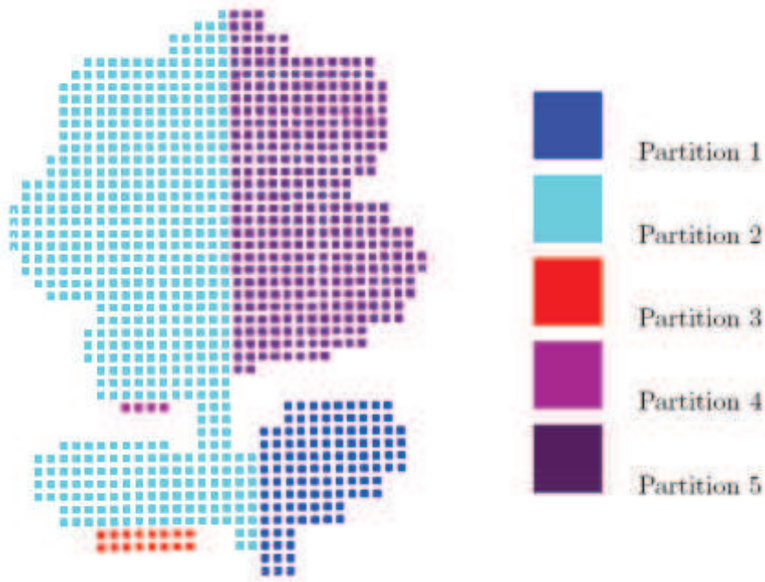


Fig. 17. The partitions of a 2D array of 1000 cells

7. Designing 3D self-assembling systems

To adapt the design algorithm so far presented for the design of 3D systems is straightforward. To expand the analysis presented in section 3 to consider the convergence of 3D automata we must replace the row-major vector representation of the automata with a row-column-major vector. This analysis shows that for a 3D automata to converge each cell must determine its next-state according to the present state of at most three neighbours; one per axis.

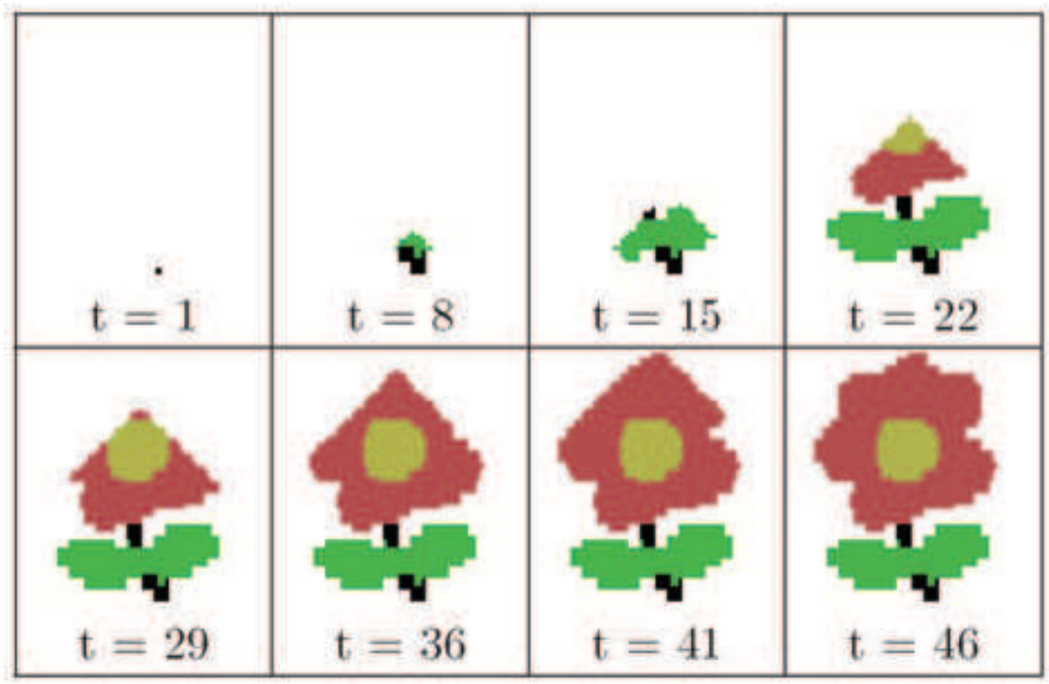


Fig. 18. The array self-assembling and converging from the origin cell

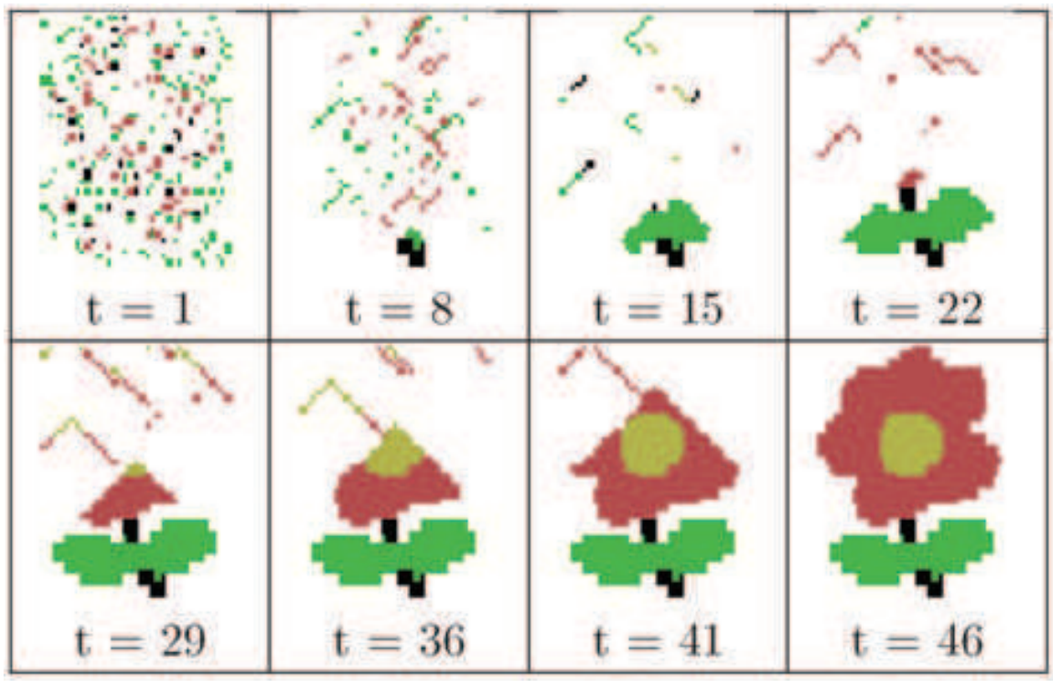


Fig. 19. The array converging from a corrupt pattern

Thus $f()$, the transition function used by each cell to determine its next-state, must be a function of three variables and the design algorithm of section must be adapted to reflect this.
Figure 20 shows the 80 partitions of a 3D irregular system of 55,000 cell. Figure 21 shows this 3D irregular system of identical cells self-assemble into the desired form. Figure 22 shows the same system repair itself after being corrupted.
Implementing the algorithm of section 6.1 using a python script, the derivation for this 55,000 cell 3D self-assembling system took 12 hours on a 3.2GHz Intel Xeon processor.

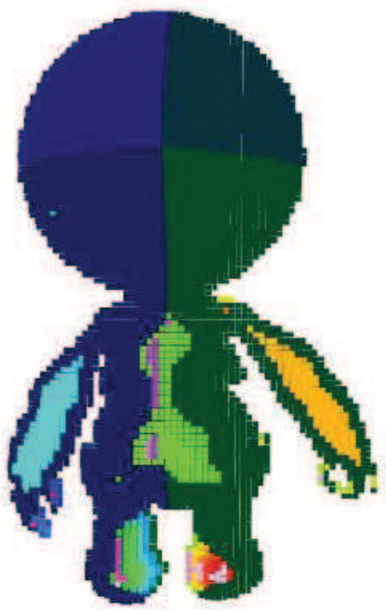


Fig. 20. The 80 partitions of a 3D robot shape, each colour is a different partition

The design required 31,637 rules and 4692 states. This compares with the 55,000 that would be required for a cartesian design (suitably adapted to cope with irregular shapes). As there is little evidence of hierarchy in the design, a hierarchical map would offer little improvement over a cartesian design. The successes of the adaptive euler-solver and evolutionary algorithms to small self-assembling systems design do not easily scale to larger systems. Their application to the design of a 55,000 cell system would require prohibitive computational resources.

8. Conclusions

During the self-assembly of a system of identical cells, each cell independently determines its state and location within the system based on local interactions with neighbouring cells and an algorithm common to each cell.
Various schemes for this algorithm have been investigated. A co-ordinate scheme; each cell determining its location from the location of its neighbours then using a map to determine where and what type of cell it should be - works for small systems but the size of the map each cell must store is impractical for larger systems. Computational genetic algorithms have been successfully used to create more efficient self-assembling solutions, but only for small simple systems. Thus an alternative is needed for large complicated systems.

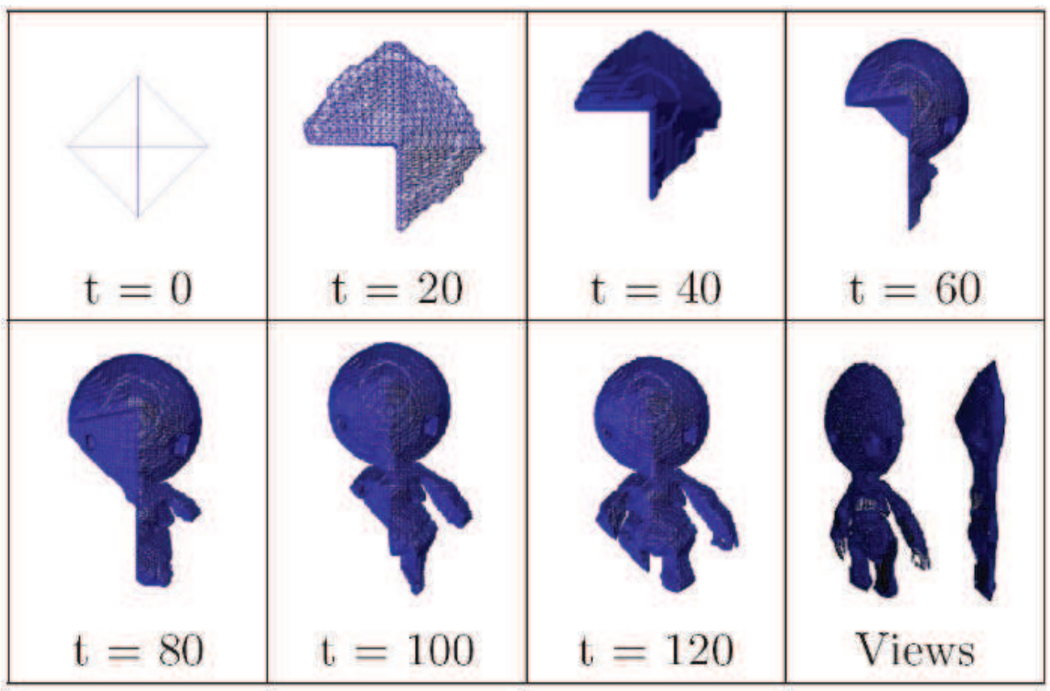


Fig. 21. A 3D system of 55,000 cells self-assembling from the origin cell

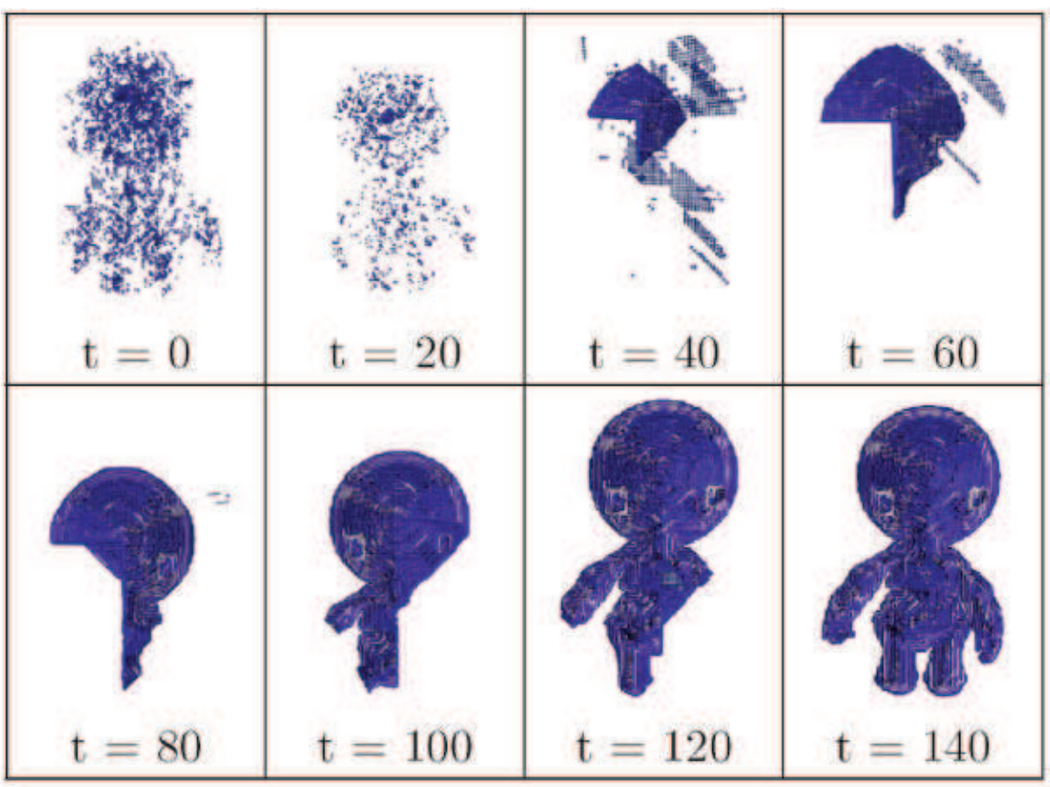


Fig. 22. The same 3D system self-healing from a corrupt shape

In this chapter we have presented a new, deterministic algorithm that can design self-assembling, self-repairing systems in two and three dimensions. The solutions generated by this design algorithm are often more efficient than those based on a co-ordinate scheme and capable of forming larger, more complicated systems than those designed by genetic algorithms.

This research has a broader application potential than just self-assembling robotics. Here we list a few of the fields where morphogenesis-inspired self-assembly may have applications:

Self-assembling micro and nano systems. Attempts to engineer biological, chemical and nano systems to self-assemble to a particular form have been limited by the difficulties of creating specific modules. If overcome, the limited complexity of each module will restrict any attempts to implement complicated self-assembly algorithms. Thus the bio-inspired minimalist strategy presented here may be an appropriate scheme.

Self-assembling micro-electromechanical systems (MEMS). Attempts to engineer MEMS to self-assemble are currently limited by the difficulties of manufacturing the complicated locomotive and inter-module bonding mechanisms. However this field is making progress in the development of small, easy to manufacture components capable of locomotion and gripping. As smaller components become possible, the emphasis on the self-assembly algorithms will be on simple schemes capable of assembling large numbers of components.

Image processing and compression. Images can be encoded as a self-assembling cellular automata described by its next-state rule. Analysis of this rule provides insight into repetition and feature scale within the image. For some images, especially images which contain repeating pattern, this next-state rule is smaller than the image bitmap - providing a means for losslessly compressing an image.

Distributed computing. Computing systems formed from large collections of processing elements are particularly difficult to co-ordinate and write software for. Morphogenesis-inspired software may provide a means of self-organising these systems, be they supercomputers or smart dust.

Plastic electronics. These are electronics that no longer reside on standard FR4 composite PCBs. Instead their components are laid on flexible substrates that are prone to stretching and ripping. One of the few remaining hurdles to the commercialisation of this technology is reliability - billions of plastic radio-frequency identification (RFID) tags cannot be printed if ten percent will fail, nor can large flexible LCD screens be rolled up if the system cannot withstand the stretching of the substrate. Morphogenesis-inspired reliability engineering may be one means of overcoming this hurdle.

Space technology. Since 2002 NASA have been running a series of workshops under the title "Ultra reliability"; this with the goal of increasing systems reliability by an order of magnitude across complex systems, hardware (including aircraft, satellites and launch vehicles) and software (Shapiro, 2006). It is not difficult to see the challenges that standard redundancy techniques will face in long life missions that are vulnerable to cosmic rays.

Heat-resistant processors. Failure rates of electronic systems increase exponentially with temperature, so perhaps morphogenesis-inspired reliability could enable computer processors to run without needing cooling fans.

Multi-agent techniques. A popular technique for modelling complex systems in software, multi-agent systems study the global effects of locally-interacting agents. Where some global effect is required of the system, it is normally very difficult to design appropriate rules of the local behaviour of the agents. We have limited this current research to the domain of a

specific type of multi-agent system, cellular automata, but it may prove applicable to the larger domain.

9. Acknowledgements

The authors would like to acknowledge the support of the EPSRC for funding the work presented in this chapter.

10. References

Barzel, R. (1992). A structured approach to physically-based modeling, *Academic Press, Cambridge MA* .

Benyus, J. M. (1998). *Biomimicry: Innovation inspired by nature*, Perennial.

Berrill, N. J. & Cohen, A. (1936). Regeneration in clavellina lepadiformis, *Journal of experimental biology* 13.

Blowey, J. (2007). Lecture notes for course: Mathematical biosciences.

Canham, R. & Tyrrell, A. M. (2003). An embryonic array with improved efficiency and fault tolerance, *Evolvable Hardware 2003* pp. 91–100.

Childress, S. (2005). Case study 2: Turing's model of chemical morphogenesis.

Eggenberger, P. (1997). Evolving morphologies of simulated 3d organisms based on differential gene expression, *Proceedings of the 4th European Conference on Artificial Life* .

Fleischer, K. & Barr, A. (1993). A simulation testbed for the study of multicellular development: multiple mechanisms of morphogenesis, *Artificial life III* .

Griffiths, A. J. F. (1976). *An introduction to genetic analysis*, W. H. Freeman and Company.

Gurdon, J. (1968). Changes in somatic cell nuclei inserted into growing and maturing amphibian oocytes, *J. Embryol. Exp. Morphol.* (20:401-14).

Jones, D. H., McWilliam, R. P. & Purvis, A. (2008). Designing convergent cellular automata, *Biosystems* .

Kizotaka, I., Watanabe, K., Tamura, H. & Ikeda, Y. (1999). Initial configuration dependence in a self-organizing robot, *Artificial life and robotics* .

Meinhardt, H. (1982). *Models of biological pattern formation*, Academic Press, London.

Miller, J. & Banzhaf, W. (2003). Evolving the program for a cell: From french flags to boolean circuits, *On Growth, Form and Computers* .

Morgan, T. H. (1904). An attempt to analyse the phenomena of polarity in tubularia, *Journal of experimental Zoology* 1: 587–591.

Murata, S., Kurokawa, H., Tomita, K. & Kokaji, S. (1999). Self-assembly and self-repair method for a distributed mechanical system, *IEEE Transactions on Robotics and Automation* .

Pisarev A., Poustelnikova E., Samsonova M. & Reinitz J. (2008) FlyEx, the quantitative atlas on segmentation gene expression at cellular resolution. Nucl. Acids Res.; doi: 10.1093/nar/gkn717

Rosee, A. L., Hader, T., Taubert, H., Rivera-Pomar, R. & Jackle, H. (1997). Mechanism and bicoid-dependent control of hairy stripe 7 expression in the posterior region of the drosophila embryo, *The Embryology journal* 16.

Shapiro, A. (2006). Ultra-reliability at nasa, *44th AIAA Aerospace Sciences Meeting and Exhibit* .

Turing, A. (1950). The chemical basis of morphogenesis, *Philos, Trans. Roy. Soc.* Ser. B 237, 37.

Yim, M., Shen, W., Salemi, B., Rus, D., Moll, M., Lipson, H., Klavins, E. & Chirikjian, G. S. (2007). Grand challenges of robotics, *IEEE Robotics and Automation* .

Cellular Automata - Innovative Modelling for Science and Engineering

Edited by Dr. Alejandro Salcido

ISBN 978-953-307-172-5
Hard cover, 426 pages
Publisher InTech
Published online 11, April, 2011
Published in print edition April, 2011

Modelling and simulation are disciplines of major importance for science and engineering. There is no science without models, and simulation has nowadays become a very useful tool, sometimes unavoidable, for development of both science and engineering. The main attractive feature of cellular automata is that, in spite of their conceptual simplicity which allows an easiness of implementation for computer simulation, as a detailed and complete mathematical analysis in principle, they are able to exhibit a wide variety of amazingly complex behaviour. This feature of cellular automata has attracted the researchers' attention from a wide variety of divergent fields of the exact disciplines of science and engineering, but also of the social sciences, and sometimes beyond. The collective complex behaviour of numerous systems, which emerge from the interaction of a multitude of simple individuals, is being conveniently modelled and simulated with cellular automata for very different purposes. In this book, a number of innovative applications of cellular automata models in the fields of Quantum Computing, Materials Science, Cryptography and Coding, and Robotics and Image Processing are presented.

InTech Europe
University Campus STeP Ri
Slavka Krautzeka 83/A
51000 Rijeka, Croatia
Phone: +385 (51) 770 447
Fax: +385 (51) 686 166
www.intechopen.com

InTech China
Unit 405, Office Block, Hotel Equatorial Shanghai
No.65, Yan An Road (West), Shanghai, 200040, China
中国上海市延安西路65号上海国际贵都大饭店办公楼405单元
Phone: +86-21-62489820
Fax: +86-21-62489821

IntechOpen