On Self-referential Shape Replication in Robust Aerospace Vehicles

Mikhail Prokopenko and Peter Wang
Centre for Intelligent Systems Design, CSIRO Information and Communication Technologies Centre
Locked bag 17, North Ryde 1670, Australia, {mikhail.prokopenko, peter.wang}@csiro.au

Abstract

We describe a multi-cellular shape replication mechanism implemented in a sensing and communication network, motivated by robust self-monitoring and self-repairing aerospace vehicles. In particular, we propose a self-referential representation (a "genome"), enabling self-inspection and self-repair; an algorithm solving the problem for connected and disconnected shapes; and a robust algorithm recovering from possible errors in the "genome". The presented mechanism can replicate combinations of predefined shapes and arbitrary shapes that self-organise in response to occurring damage.

Introduction

NASA's goal of robust aerospace vehicles requires structures that are capable of self-assessment and self-repair. Previous work in the joint CSIRO-NASA Ageless Aerospace Vehicle (AAV) project developed and examined concepts for integrated sensing and communication networks which are expected to detect and react to impact location and damage over a wide range of impact energies, ranging from micro-particles to meteoroids (Price et al., 2003; Prokopenko et al., 2004; Lovatt et al., 2003). One of the most important design principles distinguishing an intelligent vehicle health management system from other sensing systems, is the requirement for continued functionality in the presence of damage, and, ultimately, the ability to carry out repairs.

In this paper we investigate a possible first step towards the self-repairing ability, focussing, in particular, on the need for a robust self-replication of multi-cellular shapes. This *shape replication* ability should cover both "standard" and "non-standard" shapes. In other words, we expect that not only a standard shape predefined by an available structural "blueprint" can be produced when required, but also that any non-standard and unpredictable shape covering a damaged region can be dynamically replicated on demand. Importantly, we investigate the self-replication mechanism that would allow us to combine "standard" and "non-standard" shapes if necessary. Repair actions, such as shape replication, might be progressing in the environment where further impacts are likely to occur, and therefore, there is a need for *robust* shape replication algorithms.

The next section will briefly describe the notion of emergent impact boundaries, used to uniquely encode a shape that might be replicated. We follow by setting the relevant background on self-replication architectures. An algorithm for shape replication is then presented and illustrated. The algorithm incorporates a self-referential representation, and solves the problem even if the shape is "disconnected" in certain sense. Finally, we consider the case when shape replication progresses in adverse circumstances, and new impacts damage some parts of the shape being replicated.

Stable Impact Boundaries

The developed hybrid Concept Demonstrator models a two-dimensional array of cells: some cells existing in dedicated hardware (two micro-processors per cell) and some residing within inter-connected personal computers (a number of cells per PC) (Price et al., 2003). We also used a stand-alone AAV Simulator capable of simulating some simple environmental effects such as particle impacts of various energies. In the AAV Simulator, cells are represented as objects (squares) on a two-dimensional plane (e.g., Figure 1), where they asynchronously interact *only* with their immediate neighbours in von Neumann neighbourhood, through connected (geometrically overlapping) communication ports. This approach uses the idea of localised algorithms, in which simple local behaviours lead to *self-organisation* of spatiotemporal multi-cellular patterns, achieving a desired global objective.

Typically, the damage on the AAV skin caused by an impact is most severe at the point of impact (an epicentre). However, not only the cells at the epicentre are destroyed, but the communication capability of the neighbouring cells is reduced. Multiple impacts result in overlapping damaged *impact-surrounding regions* with quite complex shapes.

Let us briefly describe multi-cellular *impact boundaries*, self-organising in presence of cell failures and connectivity disruptions. On the one hand, it is desirable that an impact boundary, enclosing damaged areas, forms a continuously connected closed circuit. This circuit may serve as a reliable communication pathway around the impact-surrounding region within which communications are compromised. Every cell on a continuously connected closed circuit must always have two and only two neighbour cells, designated as the circuit members (circuit-neighbours of this cell). On the other

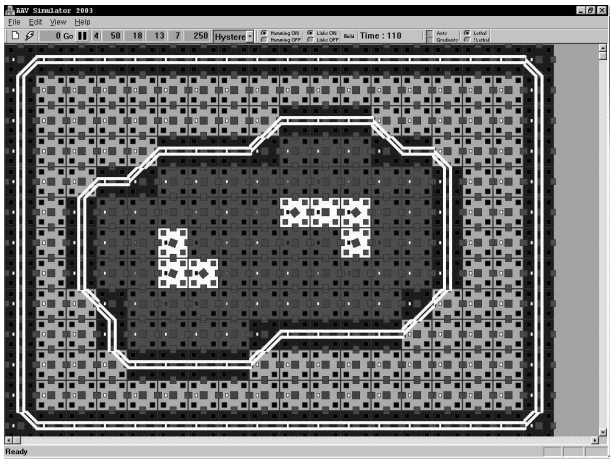

Figure 1: White cells are destroyed, dark-grey cells form "scaffolding", black cells form the "frame". Boundary links are shown as white double-lines.

hand, a continuously connected closed impact boundary provides a template for repair of the impact-surrounding region, uniquely describing its shape (Figure 1). Both these functionalities of impact boundaries can be contrasted with non-continuous "guard walls" investigated by Durbeck and Macias (Durbeck and Macias, 2002) that simply isolate faulty regions of the Cell Matrix, without connecting elements of a "guard wall" in a circuit.

In order to serve either as a communication pathway or a repair template, an impact boundary should be robust to communication failures caused by proximity to the epicentre. The algorithm producing such circuits and the metrics quantitatively measuring their spatiotemporal stability are described elsewhere (Foreman et al., 2003). In this paper, we assume that our impact boundary is a stable continuously connected closed circuit. It is sufficient to mention here the following spatial self-organising layers:

- *scaffolding* region, containing the cells that suffered significant communication damage;
- *frame boundary* — an inner layer of normal cells that are able to communicate reliably among themselves;
- *closed impact boundary*, connecting the cells on the frame boundary into a continuous closed circuit by identifying their circuit-neighbours.

The "frame" separates the scaffolding region from the cells that are able to communicate to their normal functional capacity. These internal layers (scaffolding, frame and closed boundary) completely define an impact-surrounding region as a layered spatial hierarchy. In general, the impact-surrounding region can be seen as an example of annular spatial sorting: "forming a cluster of one class of objects and surrounding it with annular bands of the other classes, each band containing objects of only one type" (Holland and Melhuish, 1999). It could be argued that, as an emergent structure, the impact-surrounding region has unique higher-order properties, such as having *an inside* and *an outside*.

Self-replication: background and motivation

In this section we attempt to position our shape replication scheme with respect to some well-known approaches. Theoretical foundations for artificial self-replicating systems were laid down by Von Neumann, who proposed two central elements: a Universal Computer and a Universal Constructor. A program Π encoded in the Universal Computer directs the behavior of the Universal Constructor. The latter is used to manufacture both another Universal Computer and a Universal Constructor. The program Π is then copied into the newly manufactured Universal Computer. It is possible to develop self-replicating automata which do not require universality. For example, a well-known self-replicating structure is a Langton's loop constructed in two-dimensional, cellular space. The loop is a closed data path, capable of transmitting data in the form of signals. These signals not only encode the loop's "genome", but serve also as the instructions for replication. In executing the instructions the loop extends itself and folds into a daughter loop, also containing the genome and capable of self-replication (Langton, 1984).

As pointed out by (Sipper, 1998), Langton's loop and its various extensions as well as other self-replicating automata based on Von Neumann architecture, can be thought of as unicellular organisms: there is a single genome describing and contained within the entire automaton. Another class of self-replicating automata includes artificial *multicellular organisms*, where each of the several cells comprising the organism contains a copy of the complete genome. One well-advanced approach exploiting such artificial multicellular organisms is the embryonic electronics (*embryonics*), aimed at very large scale integrated circuits with self-repair and self-replication capabilities (Sipper et al., 1997; Mange et al., 2000). Essentially, embryonics employs three biologically inspired principles: multi-cellular organisation, cellular differentiation, and cellular division. Cellular differentiation takes place by having each artificial cell compute its coordinates (i.e., position) within a one- or two-dimensional space, after which it can extract the specific gene within the artificial genome responsible for the cell's functionality. Cellular division occurs when a "mother cell" arbitrarily placed within the grid, multiplies to form a new multi-cellular organism. In addition to self-replication of the original circuit in case of a major fault, this artificial organism also exhibits self-repair capabilities, allowing partial reconstruction in case of a minor fault. In summary, the embryonics approach models multicellular organisms that *ontogenetically* develop in order to perform useful tasks.

Another relevant concept is *self-inspection*. In some cases, the genome is predetermined and simply needs to be replicated. This would be the case when a "blueprint" of a standard shape is available. Sometimes, however, there is a need to dynamically construct the genome describing a non-standard shape for its subsequent replication. Moreover, sometimes self-inspection should proceed concurrently with the interpretation of the genome (Laing, 1977).

The shape replication algorithms developed in the context of AAV and presented in the following sections are based on the principles of multi-cellular organisation, cellular differentiation, and cellular division as well — similarly to the embryonics approach. A desired shape is encoded when an emergent impact boundary self-inspects itself and stores the "genome" in a "mother" cell. The genome contains both data describing the boundary and a program of how to interpret these data. The mother cell is then seeded in a new place outside the affected AAV array. Executing its program initiates *cell-replication* in the directions encoded in the genome. Each cell-replication step involves copying of the genome (both data and the program) followed by differentiation of the data: an appropriate shift of certain coordinates. Newly produced cells are capable of cellular division, continuing the process until the encoded shape is constructed.

In order to provide a unifying view on the inter-related concepts briefly described above, we informally characterise the shape replication process in self-referential terms, employing two logical levels: an object level and a meta-level. It is well- known that self-replication of a system can be characterised by emergent behaviour and *tangled hierarchies* exhibiting Strange Loops: "an interaction between levels in which the top level reaches back down towards the bottom level and influences it, while at the same time being itself determined by the bottom level" (Hofstadter, 1989). In terms of shape replication, one may argue that the genome encodes the shape in each cell together with the meta-level instructions of how to replicate it. In other words, each cell contains a model of the whole multi-cellular shape, unfolding it at every cell-replication and differentiation step. In addition, we shall illustrate that self-inspection of an emergent impact boundary can be mirrored by self-inspection of the genome inside each cell, at every cell-replication step. Similarly, we shall demonstrate that self-repair of the overall damaged impact-surrounding region can be reflected in self-repair of the code embedded in each cell.

Self-referential Shape Replication

Shape Structure

In this section we provide formal definitions of an impact boundary and internal scaffolding, and draw a clear distinction between "connected" and "disconnected" cases. A two-dimensional AAV array can be represented by a *planar grid graph* $G(V,E)$: the product of path graphs on m and n vertices, where the vertices $V(G)$ are any set of points on the planar integer lattice. The edges $E(G)$ connect vertices at unit distances. Given an impact, all cells that are located within the impact-surrounding region can be represented by an *impact* subgraph A of G (Figure 2).

First of all, we identify all the vertices $S \subseteq A$ which have precisely 4 edges each. Then we define the *scaffolding subgraph* Υ_A of A as a subgraph induced on the impact graph A by the set S: i.e., as the set of the vertices S together with any edges $E(A)$ of the impact graph A whose endpoints are both in the subset S.

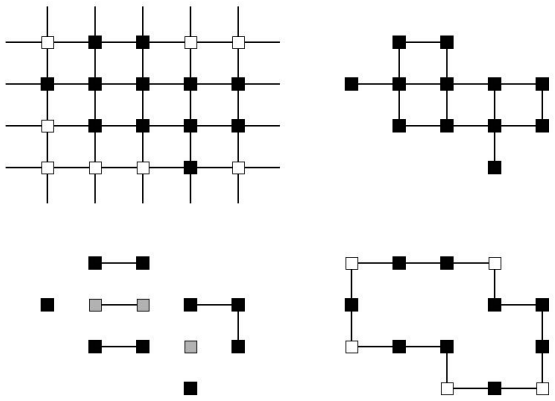

Figure 2: Top-left: a planar grid graph G, where vertices shown in black represent cells affected by an impact. Top-right: the impact subgraph A of G. Bottom-left: the scaffolding subgraph Υ_A with vertices shown in grey, and the frame-boundary subgraph Φ_A with vertices shown in black. Bottom-right: the closed-boundary subgraph Ω_A with vertices added to the frame shown in white.

Secondly, we identify the *frame-boundary subgraph* Φ_A of A as a subgraph induced on the impact graph A by the set-complement $A \setminus \Upsilon_A$: i.e., as the set of the vertices $A \setminus \Upsilon_A$ together with any edges $E(A)$ of the impact graph A whose endpoints are both in the subset $A \setminus \Upsilon_A$. Figure 2 (bottom-left) illustrates the case when both the scaffolding subgraph Υ_A and frame-boundary subgraph Φ_A are disconnected.

Finally, the *closed-boundary subgraph* Ω_A of G is defined as follows. We intend to add to the frame-boundary subgraph precisely those elements from G which provide a shortest path (outside the scaffolding subgraph) between components of the possibly disconnected frame-boundary subgraph. Formally, we employ the graph-theoretic definition of convex sets, according to which a set of vertices X in a connected graph is called convex if for every two vertices $u, v \in X$, the vertex set of *every shortest path* between u and v lies completely in X. We now identify the *graph-theoretic convex hull* H_A of the set of frame-boundary vertices $V(\Phi_A)$ in G but not in scaffolding subgraph Υ_A, as the smallest graph-theoretic convex set in $V(G) \setminus V(\Upsilon_A)$ containing $V(\Phi_A)$. A subgraph induced on the graph G by the set H_A is the desired closed-boundary subgraph Ω_A (Figure 2: bottom-right). It can be shown that the closed-boundary subgraph Ω_A is always *cyclic*.

These definitions are not constructively used by the decentralised boundary formation algorithms, localised within each cell (Foreman et al., 2003). The graph-theoretic notions require global information, used, for example, in specifying all the vertices of the impact graph in advance, or finding shortest paths and convex hulls. In reality, autonomous cells asynchronously deal with unreliable communication messages, while trying to determine whether they belong to scaffolding, frame or closed boundary. We introduced here the formal definitions in order to give a graph-theoretic semantics to these emergent structures, and in particular, to distinguish between *connected* and *disconnected* scaffolding sub-

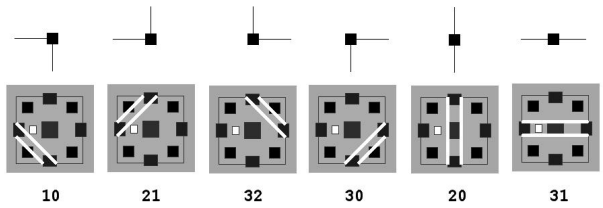

Figure 3: Boundary links.

x	0	0	0	1	2	3	3	4	4	4	3	2	2	1
y	0	1	2	2	2	2	1	1	0	-1	-1	-1	0	0
λ	32	20	30	31	31	10	32	10	20	21	31	32	10	31

Table 1: An example boundary genome.

graphs. It is precisely the second case that presents some difficulties for shape replication.

Given the planar grid topology, each cell on the closed impact boundary may have 6 boundary links, connecting ports "left-right", "left-top", etc. Enumerating four communication ports from 0 to 3 ("bottom" to "right" clockwise) allows us to uniquely label each boundary link with a two-digit number λ, e.g., "32" would encode a link between the "right" and "top" ports (Figure 3). Then, the whole impact boundary can be encoded in an ordered list of these labels. For instance, the boundary depicted in the Figure 2 can be simply represented by the list $\{32,\ 20,\ 30,\ 31,\ 31, 10, 32, 10, 20, 21, 31, 32, 10, 31\}$. However, in order to replicate the bounded shape, filling it cell by cell, we need to introduce a system of coordinates relative to a cell containing the shape list. More precisely, the boundary genome is a list of triples (α, β, λ), where (α, β) are relative coordinates of a cell with the boundary link λ. The boundary genome for our example is shown in the Table 1. The instructions of how to interpret these data can be easily represented in an assembler-like language, with each "program" triple encoding two operands and an instruction type.

An Algorithm for Disconnected Scaffolding

The first phase is *self-inspection of the impact boundary*, producing the genome, e.g., the genome in the Table 1. The process starts with a selection of a mother cell (any cell s_0 on the boundary), and involves the following steps:

- the mother cell inserts the triple $(0, 0, \lambda_0)$ into the empty genome, where λ_0 is the boundary link maintained by the cell s_0, and sends the incomplete genome to the neighbour in a specific direction (e.g., counter-clockwise);
- each boundary cell receiving the genome determines the relative (a, b) coordinates of the message sender given the port of the incoming message: e.g., if the message comes on the bottom port (labelled as 0), then the sender's relative coordinates are $(0, -1)$ (the possibilities are encoded in a look-up Table 2);
- the cell increments all (x, y) coordinates in the genome as follows: $x = x + a$, $y = y + b$;
- the cell appends the triple $(0, 0, \lambda_i)$ to the genome, where λ_i is the the boundary link maintained by this cell;
- if this cell is not the mother cell, it sends the genome to the counter-clockwise neighbour, otherwise, the process terminates.

port	0	1	2	3
a	0	-1	0	1
b	-1	0	1	0

Table 2: The look-up table of directions and coordinates.

The next phase is *shape replication per se*. It starts when the mother cell is seeded in some available space. The process involves cell-replications carried out by not only new boundary cells, but also by new scaffolding cells. The cell-replication program encoded in each cell has the following steps (starting with the seed at the beginning):

- the cell iterates through the genome and determines whether there is a triple $(0, 0, \lambda)$, for some λ;
- if such a triple is found, a Boolean flag ω is set to true ("boundary cell"); otherwise $\omega =$ false;
- the cell iterates through all possible directions π, where $0 \leq \pi \leq 3$, doing for each π the following:
  1) retrieve from the look-up Table 2 the coordinates (a, b) for construction in the direction π;
  2) check if there is a cell at the relative location (a, b): if there is, then the direction π should not be used, and the cell moves to the step (6), otherwise it continues with the following steps;
  3) check if both $\omega =$ true and the two-digit number λ does not include the digit π;
  4) if the condition (3) is satisfied (meaning a boundary cell is considering to produce a scaffolding cell), then
     a) fix the vertical "strip" $x = a$ and, by varying the y coordinate across the genome, compute the number of times n_+ and n_- the boundary fully crosses this "strip", above and below $y = b$ respectively (this computation is described below);
     b) check if either n_+ or n_- is odd and neither is 0;
  5) if either the condition (3) is not satisfied, or the conditions (3) and (4.b) are both satisfied (a boundary cell produces a scaffolding cell), then
     a) construct a cell in the direction π;
     b) copy the genome to the constructed cell;
     c) decrement all coordinates in the constructed cell's genome as follows: $x = x - a$, $y = y - b$;
  6) increment the direction π;
- the process stops when all directions have been checked.

In order to compute the numbers n_+ and n_- relative to the location (a, b), the cell iterates through the genome for $x = a$, varying $y > b$ and $y < b$ respectively. The number n_+ is initiated to 0, and is incremented each time either a) an entry $\lambda = 31$ is encountered, or b) $\lambda = 32$ is encountered, followed (not necessarily immediately) by $\lambda = 10$, or c) $\lambda = 21$ is encountered, followed (not necessarily immediately) by $\lambda = 30$. The number n_- is computed similarly with the λ pairs in (b) and (c) reversed. The numbers n_+ and n_- determine whether the location (a, b) is *inside* or *outside* the shape.

Let us exemplify the cell-replication phase, and in particular cell-differentiation occurring at step (5). We continue with the example genome (Table 1), and assume that the seed $(0, 0, 32)$ starts the process from the bottom-left corner of the boundary. Let us start with the direction $\pi = 0$. The look-up table suggests the coordinates $a = 0, b = -1$, the shift down. The location $(0, -1)$ is free. The condition (3) is satisfied as the number $\lambda = 32$ does not have $\pi = 0$ in it. However, the vertical "strip" on which a possible new cell

x	0	0	0	1	2	3	3	4	4	4	3	2	2	1
y	-1	0	1	1	1	1	0	0	-1	-2	-2	-2	-1	-1
λ	32	20	30	31	31	10	32	10	20	21	31	32	10	31

Table 3: The genome updated after cell-differentiation.

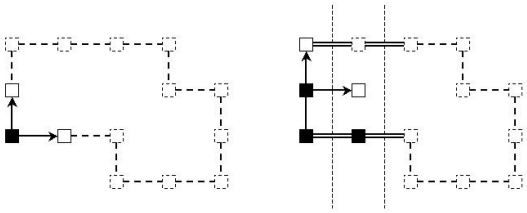

Figure 4: Shape replication. Boundary cells encoded in the genome but not yet produced are shown with dashed lines. Left: a black cell (seed) produces two white cells, indicated by arrows. Right: Two more cells are being produced: one of them is a scaffolding cell, pointed to by the horizontal arrow. The inside direction is recognised by the vertical strip being "crossed" above and below the considered location.

would be located is not "crossed" by the boundary at all, so $n_+ = n_- = 0$, and there is no need to produce a scaffolding cell (which would be outside of the desired shape). The next direction $\pi = 1$ is similar. The direction $\pi = 2$, however, "belongs" to the number $\lambda = 32$, triggering the production of another boundary cell with the coordinates $a = 0, b = 1$. A cell is constructed in the direction 2 (top) relative to the seed; the genome is copied to the newly constructed cell which is differentiated by updating all coordinates as follows: $x = x - 0$, $y = y - 1$. In other words, all y coordinates are decremented, resulting in genome shown in Table 3.

The (seed) cell that produced the copy is encoded in the copy's genome by triple $(0,-1,32)$, i.e., the seed and the copy have different representations of the same shape. The last direction $\pi = 3$ is similar, and another boundary cell is produced to the right. The shape replication process is now driven by these two newly produced cells. For example, the cell produced to the top of the seed considers 4 directions π. It excludes $\pi = 0$, because there is a cell already in the place indicated by $\pi = 0$. The direction $\pi = 1$ is excluded because it's outside of the shape, as recognised by $n_+ = 0$ and $n_- = 0$. The direction $\pi = 2$ is selected because 2 is within $\lambda = 20$, triggering the cell-replication and cell-differentiation process similar to the one described previously. The direction $\pi = 3$ is interesting now because the corresponding location $a = 1, b = 0$ is inside the shape as recognised by $n_+ = 1$ and $n_- = 1$. In other words, there are two places where the boundary "crosses" the strip $x = a = 1$: one above the level $y = b = 0$, and one below (Figure 4:right).

The process terminates precisely because boundary cells distinguish between inside and outside, and do not replicate outside the desired shape. The scaffolding cells can only reach the boundary from inside: if a scaffolding cell produces a new cell whose genome has a triple $(0,0,\lambda)$ after an update, then this replica recognises itself as a boundary cell, and does not replicate outside. If the scaffolding subgraph was always connected, a simpler algorithm would be possible. It would involve a) boundary cells building *only* other boundary cells; and b) seeding a single scaffolding cell. It would, furthermore, avoid any need to verify whether replicated cells are inside or outside the shape (i.e., the numbers n_+ and n_- would not be needed). In either case (connected or disconnected scaffolding) the shape replication is driven by multiple cells, progressing in parallel (Figure 5).

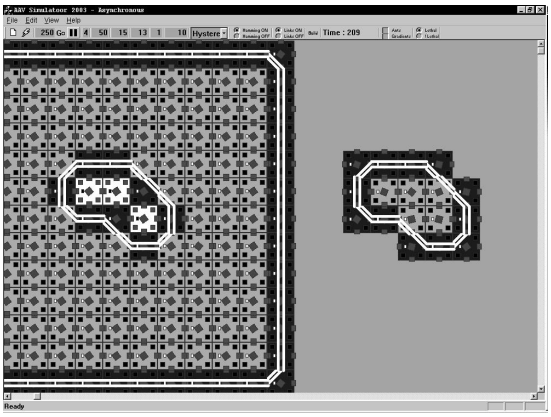

Figure 5: Completed shape replication.

Robust Shape Replication

The shape replication process described above assumes that at any cell-replication step there are no errors. However, there may be cases when some fragments of the genome are damaged due to copying process or processors/memory failures. If the genome is not repaired then the shape replication process would not terminate at the boundary, and some cells would be replicated beyond the desired shape by going through missing boundary cells.

In this section we consider an advanced robust algorithm designed to recover from such errors. More precisely, we consider the case when some triples (x,y,λ) in the genome are corrupted. The main challenge is not only to recognise corrupted data, but to avoid a replication that may produce an incorrect shape. The proposed solution involves 1) self-inspection of the genome within a cell, determining the endpoints of disconnected boundary fragments, 2) self-repair of the genome within a cell, adding the triples between the disconnected fragments. Thus, self-inspection of the genome on the cell level *mirrors* the boundary self-inspection, while self-repair of the genome on the cell level *mirrors* the repair of the overall shape. We believe that these are examples of Strange Loops because on the one hand, events on the multi-cellular shape's level trigger cellular transformations (e.g., a global repair activating the ontogenetical development), while on the other hand, the actions carried out by each cell (e.g., self-repair of the encoded genome) affect the higher level where the shape is replicated.

Each phase of *self-inspection of the genome within the cell* determines a pair of triples (x_1,y_1,λ_1) and (x_2,y_2,λ_2) such that the triple (x_1,y_1,λ_1) is the last before the break in the counter-clockwise direction, and (x_2,y_2,λ_2) is the first after the break counter-clockwise. If either of these triples is not found, then the genome damage is not repairable. Otherwise, a *self-repair* phase follows with the following steps:

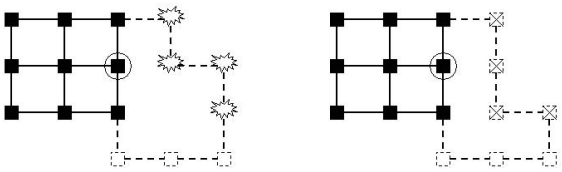

Figure 6: The cell shown inside a circle attempts self-repair. Left: the corrupted triples are shown with the "star"-like signs. Right: the repaired triples are marked with crosses.

- for the start triple (x_1, y_1, λ_1): a) select a counter-clockwise direction π_1 in λ_1; b) retrieve from the look-up Table 2 the coordinates (a_1, b_1) for repair in the direction π_1; c) set $x_1 = x_1 + a_1$ and $y_1 = y_1 + b_1$;
- for the end triple (x_2, y_2, λ_2): a) select a clockwise direction π_2 in λ_2; b) retrieve from the look-up Table 2 the coordinates (a_2, b_2) for repair in the direction π_2; c) set $x_2 = x_2 + a_2$ and $y_2 = y_2 + b_2$;
- approximate the line between the points (x_1, y_1) and (x_2, y_2) and its slope μ;
- set $x = x_1$, $y = y_1$, and continue an iterative process until the points (x, y) and (x_2, y_2) are the same:
  1) determine coordinates (a, b) such that the line between $(x + a, y + b)$ and (x_2, y_2) has a slope closest to μ; if the genome contains a triple $(x + a, y + b, \lambda^*)$ for some boundary link λ^*, then the algorithm terminates and the repair fails because a projected fragment intersects an existing boundary;
  2) retrieve from the look-up Table 2 the direction π corresponding to the coordinates (a, b);
  3) set the direction π_1^* opposite to π_1 by using modulo 4, i.e. $\pi_1^* = (\pi_1 + 2)$ *mod* 4;
  4) form λ by concatenating π and π_1^* in decreasing order;
  5) insert (x, y, λ) and set $x = x + a$, $y = y + b$, and $\pi_1 = \pi$;
- when the points (x, y) and (x_2, y_2) are the same, "seal" the break:
  1) set the direction π_2^* opposite to π_2 as $\pi_2^* = (\pi_2 + 2)$ *mod* 4, and the direction π_1^* opposite to π_1 as $\pi_1^* = (\pi_1 + 2)$ *mod* 4;
  2) form λ by concatenating π_1^* and π_2^* in decreasing order;
  3) insert (x, y, λ);

The genome is partially repaired (Figure 6) within each cell which detected a discontinuity. Although the repaired genome does not cover all the missing cells, it does not introduce any cells which were not in the original shape, exhibiting soundness but not completeness property. In other words, the repaired boundary is always contained within the original shape. Importantly, there is a redundancy in the shape replication process: other cells which did not suffer any damage would successfully replicate the parts not encoded in the partially repaired genomes.

The described algorithms handle both standard ("blueprint") and non-standard shapes, self-organising in response to damage. Moreover, it is possible to combine these types. For example, structural data can be encoded in the form of triples, and a given genome can be extended in run-time with the data produced by self-inspecting emergent boundaries. Similarly, the self-repair phase within a cell which detected an anomaly in the genome may draw some data from the structural "blueprints" rather than approximate segments between disconnected fragments.

Conclusions and Future Work

We investigated a multi-cellular shape replication mechanism, implemented in a sensing and communication AAV network. The main algorithm solves the problem for connected and disconnected scaffolding. The underlying self-referential representation enables self-inspection and self-repair — contributing to a robust algorithm that can recover from possible errors in the "genome". The presented mechanism can replicate predefined standard shapes, arbitrary emergent shapes and their combinations. One future direction would involve treating the program in the same way as the data, so that *reprogrammable* cells may redirect and improve an ongoing shape replication process.

References

Durbeck, L. and Macias, N. (2002). Defect-tolerant, fine-grained parallel testing of a cell matrix. In Schewel, J., James-Roxby, P., Schmit, H., and McHenry, J., editors, *Proceedings of SPIE ITCom 2002 Series, Vol. 4867*.

Foreman, M., Prokopenko, M., and Wang, P. (2003). Phase transitions in self-organising sensor networks. In Banzhaf, W., Christaller, T., Dittrich, P., Kim, J., and Ziegler, J., editors, *Advances in Artificial Life - Proceedings of the 7th European Conference on Artificial Life (ECAL)*, volume 2801 of *LNAI*, pages 781–791. Springer Verlag.

Hofstadter, D. R. (1989). *Gödel, Escher, Bach: An Eternal Golden Braid*. New York: Vintage Books.

Holland, O. and Melhuish, C. (1999). Stigmergy, self-organization, and sorting in collective robotics. *Artificial Life*, 5:173–202.

Laing, R. (1977). Automaton models of reproduction by self-inspection. *Journal of Theoretical Biology*, 66:437–456.

Langton, C. (1984). Self-reproduction in cellular automata. *Physica D*, 10:135–144.

Lovatt, H., Poulton, G., Price, D., Prokopenko, M., Valencia, P., and Wang, P. (2003). Self-organising impact boundaries in ageless aerospace vehicles. In Rosenschein, J., Sandholm, T., Wooldridge, M., and Yokoo, M., editors, *Proceedings of the 2nd International Joint Conference on Autonomous Agents and Multi-Agent Systems*, pages 249–256. ACM Press.

Mange, D., Sipper, M., Stauffer, A., and Tempesti, G. (2000). Towards robust integrated circuits: The embryonics approach. *Proceedings of the IEEE*, 88:516–541.

Price, D., Scott, A., Edwards, G., Batten, A., Farmer, A., Hedley, M., Johnson, M., Lewis, C., Poulton, G., Prokopenko, M., Valencia, P., and Wang, P. (2003). An integrated health monitoring system for an ageless aerospace vehicle. In *Proceedings of the Fourth International Workshop on Structural Health Monitoring*. Stanford University.

Prokopenko, M., Wang, P., Foreman, M., Valencia, P., Price, D., and Poulton, G. (2004). On connectivity of reconfigurable impact networks in ageless aerospace vehicles. *Journal of Robotics and Autonomous Systems*, Special Issue:in press.

Sipper, M. (1998). Fifty years of research on self-replication: An overview. *Artificial Life*, 4:237–257.

Sipper, M., Mange, D., and Stauffer, A. (1997). Ontogenetic hardware. *BioSystems*, 44:193–207.