METAMorph: Experimenting with Genetic Regulatory Networks for Artificial Development

Finlay Stewart, Tim Taylor, and George Konidaris

Institute of Perception, Action and Behaviour
University of Edinburgh
f.j.stewart@sms.ed.ac.uk, timt@inf.ed.ac.uk, gdk@cs.umass.edu
`http://homepages.inf.ed.ac.uk/s0091983/METAMorph/`

Abstract. We introduce METAMorph, an open source software platform for the experimental design of simulated cellular development processes using genomes encoded as genetic regulatory networks (GRNs). METAMorph allows researchers to design GRNs by hand and to visualise the resulting morphological growth process. As such, it is a tool to aid researchers in developing an understanding of the expressive properties of GRNs. We describe the software and present our preliminary observations in the form of techniques for realising some common structures.

1 Introduction

Genetic regulatory networks (GRNs) [6] have recently become a popular model of gene expression employed in genetic algorithms to study artificial developmental processes [2, 3, 1, 4]. GRNs model the natural processes of intra- and inter-cell protein signalling during gene expression. However, the dynamics produced by GRNs and the properties of the associated morphological search space are difficult for researchers designing evolutionary systems to understand. The expressive properties of any encoding used as a basis for an evolutionary process impose a bias on the kinds of solutions that process is likely to find. It is therefore difficult to understand the dynamics of evolutionary GRN models without an understanding of the kinds of shapes that GRNs are biased toward expressing.

In this paper, we present METAMorph, an open source software application that allows for the hand-design and execution of artificial genomes that express cellular growth through GRNs. METAMorph aims to provide an environment within which we can experiment with one model of a GRN in order to understand its natural dynamics and the kinds of structures it is biased toward. It is also potentially valuable as a interactive teaching tool for understanding cellular growth processes. In the following sections we describe the GRN model used in METAMorph, provide a demonstration of its use in designing the development process of a cigar-shaped organism, and present a few design techniques for creating common structures.

2 Computational Development and Genetic Regulatory Networks

The field of computational development is the study of artificial models of embryonic cellular development, with the aim of understanding how complex structures and forms can develop from a small group of seed cells [4]. The natural way to study development is through the simulation of some abstract model of the dynamics of the cellular development process, often within the context of an evolutionary process.

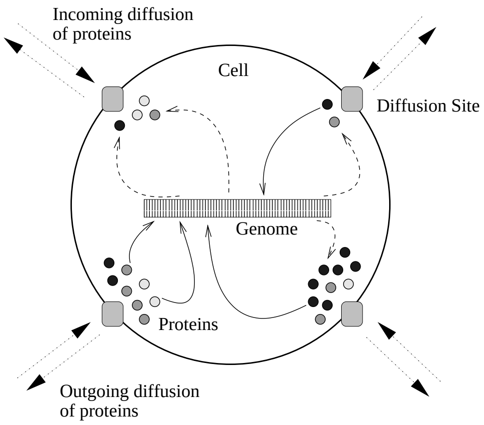

Fig. 1. Cellular GRNs. The same genome is expressed at all sites of all cells. However, local protein concentrations affect the production of new proteins at each site by enhancing and inhibiting genes. Proteins can diffuse between sites within a cell, and in some cases, between cells.

GRNs model the interaction between genes, proteins, and the cellular environment. Each cell contains the same genome, but a potentially different set of active proteins, distributed across a fixed set of diffusion sites on the cell. Each gene has a set of enhancer and inhibitor proteins, which increase or decrease its activation, respectively, and when activated, produces some protein, using some output function (which scales its protein output according to its activation) and some distribution function (which places produced proteins at the cell's diffusion sites). Protein levels are also subject to attenuation (where protein levels decay) and diffusion (where proteins at diffusion sites move to other sites, in the same cell or on adjacent cells). Proteins can also be used as sensor mechanisms (where the cell produces them by itself under certain conditions), and as actuation mechanisms (where sufficient concentrations cause a cellular event, such as cell division, to occur). These processes are summarised in Figure 1. In this way, genes form a network whereby the interaction of the elements they produce regulates further gene expression. Variants of this basic abstract idea have been used in conjunction with an evolutionary process for a variety of applications, such as simulated cellular development [2, 3, 1, 4], real-time robot control [5] and for the control of groups of underwater robots [7].

GRNs are thus a natural model for cellular development, and appear to possess desirable properties for acting as an evolutionary substrate [6] (e.g., they show a strong tendency towards modularity [1]). However, evolved GRNs are difficult for humans to understand, and we do not even have a qualitative measure of how difficult some natural structures are to achieve using them. In the following section, we introduce METAMorph, which aims to facilitate the accumulation of such knowledge through experimentation.

3 METAMorph

In the METAMorph (Model for Experimentation and Teaching in Artificial Morphogenesis) framework, multicellular artificial organisms are grown from a single cell (the *zygote*) using GRNs, with some subset of the proteins produced being able to trigger cell-level actions such as cell division or death. Many details are omitted for brevity; the interested reader is referred to the URL given above.

3.1 Proteins

A protein is defined by a unique name, a type (internal or external) and two constants: decay (τ) and diffusion (λ). Internal proteins may only diffuse within a cell, whereas external proteins pass through the cell membrane and can hence be used for inter-cellular signaling.

Internal proteins The proportion of the protein that is lost due to decay at each timestep is specified by τ:

$$conc_{prot}(t+1) = (1-\tau)conc_{prot}(t)$$

The concentration of each protein at 12 sites around the cell is stored; thus proteins may be unevenly distributed within the cytoplasm. The genome is expressed separately at each of these sub-cellular sites based on the local protein concentrations. The diffusion constant specifies the proportion of the protein that diffuses to neighbouring sites at each timestep. Due to the isospatial layout of the sites (see Figure 2), each one has exactly four equidistant neighbours between which this diffused protein is equally shared.

$$conc_{prot}(p, t+1) = (1-\lambda)conc_{prot}(p, t) + \sum_q neighbour(p, q)\frac{\lambda}{4}conc_{prot}(q, t)$$

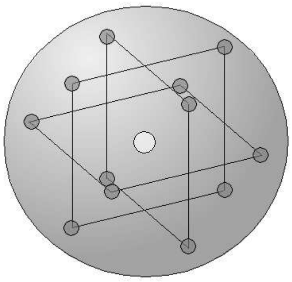

Fig. 2. The 12 protein sites (filled circles) are located at the corners of three mutually orthogonal squares centred on the centre of the cell (open circle).

External proteins Concentrations of external proteins are represented by isotropic 3D Gaussian distributions centred on the cell-site at which the protein originates. The diffusion constant specifies how much is added to the variance at each timestep. The decay constant determines how much the total concentration (i.e. the integral of the Gaussian function) should be reduced at each timestep. Note that external proteins diffuse freely through cells.

3.2 Genes

All cells have the same genome comprising a number of genes. Each gene produces exactly one protein, although the same protein may be produced by several genes. The amount produced by the gene depends on zero or more promoter sequences attached to that gene. Each promoter sequence consists of a protein name and a weighting. Thus a weighted sum of protein concentrations at that site is calculated:

$$a_{prot} = \sum_{prot}^{promoterseqs} weight_{prot} conc_{prot}(x, t)$$

This value is the input for a sigmoid function with a certain bias and steepness defined in the gene:

$$conc_{prot}(x, t+1) = conc_{prot}(x, t) + \frac{1}{1 + e^{-steepness(a_{prot} - bias)}}$$

3.3 Cells

Cells are represented as spheres of set radius. They each occupy a position on an isospatial grid in three dimensions (the same geometry as is used for the sites within cells), meaning that each cell can potentially have 12 equidistant neighbours. Cells can perform a number of actions, with an action being triggered when a specific protein's mean concentration in the cell exceeds a threshold value. These actions are as follows:

- **Cell division** When a cell divides it produces a daughter cell in the adjacent grid space in the direction of the mitotic spindle (see below), as long as that space is vacant. Cytoplasmic proteins may be shared unequally between mother and daughter, as the spatial distribution of proteins in the cell is taken into account during the split.
- **Mitotic spindle movement** Each cell has a 'mitotic spindle' that points in one of the 12 grid directions at any given time and defines the direction in which cell division takes place. This spindle may be moved forwards or backwards one step along an equatorial line around the cell as a result of a protein threshold being reached. Another action changes which 'orbit' the spindle is on. Alternatively, the spindle can be made to point in the direction of the sub-site where the concentration of a given protein is either highest or lowest.
- **Programmed cell death (apoptosis)** The cell is removed from the world leaving a vacant grid space.
- **Differentiation** Cells can have various different types. The type of a cell has no effect on its function, but is visualised by its colour. This feature is included to allow the investigation of how heterogeneous organisms can be created, e.g. in animals cells specialise as skin cells, blood cells, neurons, etc.

4 An Example Organism

This section describes how a cigar-shaped organism consisting of around 700 cells can be made. This is a fairly simple shape to create, as it is radially symmetric in all dimensions but one. It is included to give the reader a flavour of the techniques that can be used in morphogenetic design. A more detailed account (and downloadable demo) is available online, along with examples of more complex shapes.

The basic method of building the shape is as follows. A line of approximately 10 cells is created, which will form the central 'axis' of the cigar (Figure 3(a)). These cells all emit an external protein (Figure 3(b)). Cells will then grow wherever the concentration of this signal protein is above a certain level, which will result in a cigar shape being produced (Figure 3(c)).

To make cells proliferate in all directions, we can set up a genome where proteins that trigger division, spindle movements and orbit switches (let us call them *split*, *turn* and *switch*, respectively) have genes with thresholds ≤ 0, i.e. they will be expressed unless actively inhibited. An unchecked cell will therefore frequently move its spindle, switch the spindle's orbit, and divide. However, this behaviour must be prevented in some situations or we will just have an ever-growing ball of cells.

The cells comprising the axis are functionally different from the rest, as they must emit an external protein (call it *signal*). We can code this distinction by the expression of a protein: cells with a high concentration of *axis* belong to the axis. *Axis* can then enhance *signal*. It can also inhibit *turn*, since we require the axis to grow in a straight line. Finally, by making *axis* an enhancer for itself, we can lock cells in the axis state once the level of *axis* exceeds some threshold.

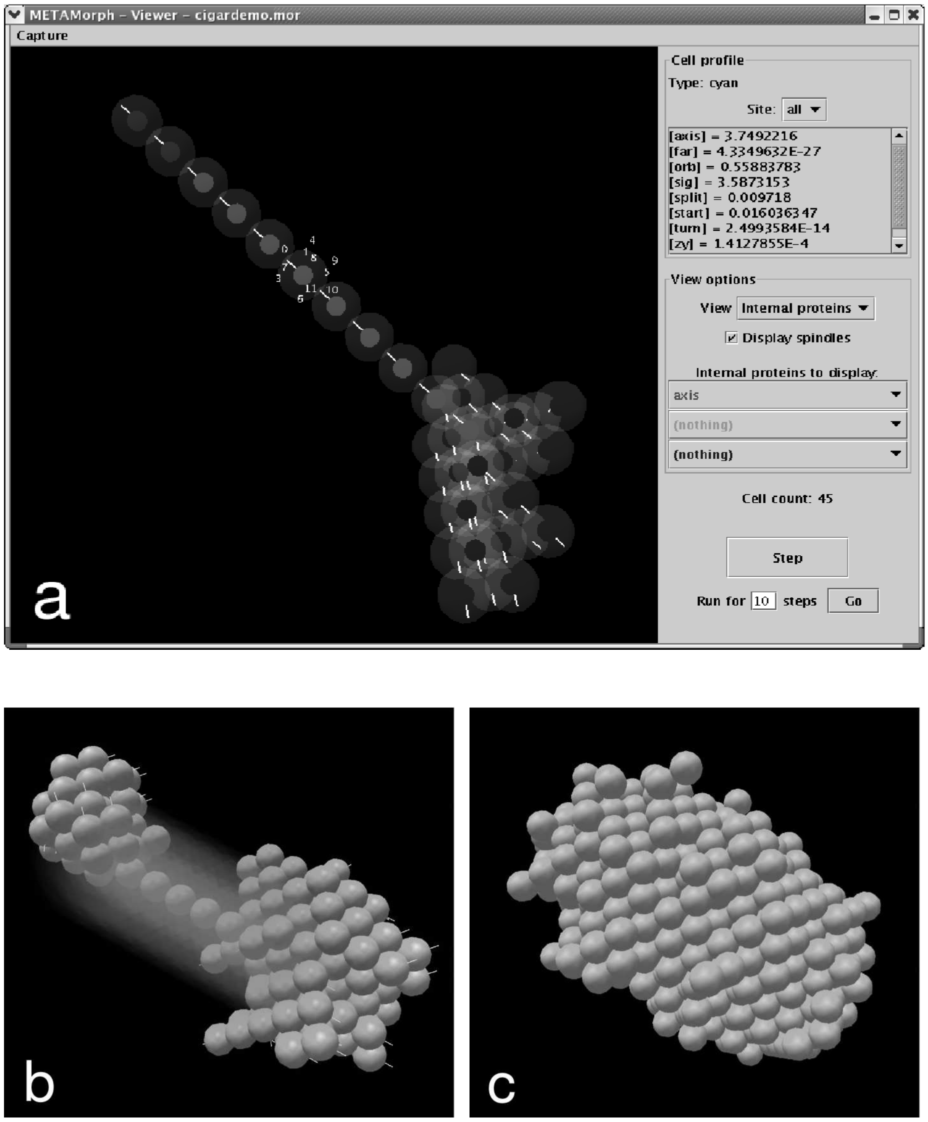

Fig. 3. Development of the cigar embryo. (a) Early in development, showing the METAMorph GUI. The light-centred cells are those with a high concentration of *axis*. The lines show the orientation of each cell's mitotic spindle; note that the axis cells' spindles are aligned as *axis* inhibits *turn*. Cells have started to proliferate from the zygote end of the axis (bottom right). (b) Cells are now growing from both ends. The cloud surrounding the axis represents the concentration of external protein *signal*. (c) The completed cigar, consisting of 684 cells. Note the existence of some offshoot cells which will soon die due to excessive levels of *far*.

To limit the length of the axis, we ensure (by a process of trial and error) that the cells divide slightly more frequently (sharing their *axis*) than the autocatalytic nature of *axis* is able to replenish. Thus *axis* will become diluted with each division until it dips below the threshold needed to sustain itself. This cell will therefore become 'ordinary', i.e. it will not express *signal*, but will move its spindle, as will its daughters. In fact, by careful initial placing of proteins in the zygote, we can make the first cell become ordinary too, allowing the cigar to be 'filled out' from both ends.

All that remains to be explained is how we limit the outward growth of these cells to produce the desired shape. For this we need a protein (which we shall call *far*) that triggers apoptosis. It is inhibited by *signal*, so when the concentration of signal becomes too low (because the cell is far from the axis), *far* will accumulate and kill the cell. The organism therefore never reaches a stable configuration, but continually grows and kills cells at its periphery. Note that *axis* (or some other initially plentiful protein) must inhibit *far*, as there is no *signal* gradient set up at the beginning of the simulation.

One can now see how a few simple changes could be made to alter the eventual form of the organism. To make the cigar thicker, for example, one would simply set the threshold of the *far* gene higher, meaning that less *signal* would be needed to keep cells alive. Making the cigar longer is a little more complex, but could be achieved by setting the threshold of *axis*'s self-enhancing gene lower so that the cells of the spine could sustain more splits before the concentration of axis was too low to sustain the process.

To give an indication of the complexity of the algorithm, it requires 55 timesteps to reach a size of 680 cells, which takes approximately 7 minutes on a PC with a 2.4GHz Athlon processor.

While there is nothing very conceptually difficult about designing organisms in METAMorph, we have found that it typically involves a considerable amount of time-consuming trial and error. This is due to the inherently parallel nature of the morphogenetic processes. GRNs can suffer from a deleterious 'butterfly effect' whereby a seemingly innocuous change can have large unforeseen effects. For these reasons, designing a more complex 'quadruped' shape, consisting of a body with four appropriately sized and positioned limbs (see website), took one of us around a week.

Although our intention in this work was to investigate morphogentic processes by hand-designing organisms, METAMorph could be easily adapted to evolutionary experiments by adding code to generate candidate genomes and assess the resultant organisms according to some fitness function.

5 General Techniques

In this section we discuss a number of principles and techniques in morphogenetic 'programming', based on our preliminary experiments attempting to constuct various shapes. Some may exploit peculiarities in our model, but it is our hope that some may represent general mechanisms for morphogenesis, applicable to other multicellular GRN models, and perhaps even having analogues in biology.

Growth in all directions, with boundary control by external protein gradient. This technique for generation and maintenance of form is described in section 4. It is robust due to its dynamic nature, as legitimate cells that are killed for any reason will grow back. Furthermore, the shape of the embryo can be easily altered *during a simulation* by changing the distribution of the signal protein.

Functional differentiation of cells. Since all cells share the same genome, it can be difficult to make some subgroup behave differently to the others. A good way to achieve this effect is to use a 'marker' protein (e.g. *axis* in the cigar example) whereby cells which have a non-negligible level of this protein will behave one way, the rest another. By making this protein autocatalytic (i.e. an enhancer for a gene producing itself) and using a substantially positive bias, this differentiation can be made to last indefinitely while still allowing cells to be switched either way by the use of other promoters. Note that this mechanism means that cell function will generally be inherited when a cell divides, which may or may not be desirable depending on the situation.

Delays. In some situations it can be useful for a cell to wait for a period of time before initiating an action. For instance, in creating a hollow shape, we may want to give cells a chance to grow outwards before killing off those that are too close to the centre. A way to achieve this effect is to create an 'accumulator' protein with a low decay constant. A fully-activated gene produces the accumulator for a number of time-steps, then another protein is produced when the accumulator finally reaches some threshold. Even longer delays can be created by using a chain of accumulators.

Quasi-binary processing. Protein concentrations and gene activations are continuous. While this is potentially useful, in many situations we require a sharp cut-off between states; e.g. a cell's membership of the cigar's 'axis' is a binary variable. A first step towards achieving this kind of behaviour is to use high steepness values for genes. Typically around 90% of the genes in our programs have step-like activation functions (steepness > 10). A typical situation is that where we have a continuously varying concentration of a 'primary' protein (often an external one), with a 'secondary' protein being produced in an all-or-nothing manner when its concentration is above or below a certain threshold level.

If we set the decay constant of this secondary protein high, its concentration at any time will depend chiefly on whether its gene was active on the last time-step (assuming that only one gene produces the protein). In this way, the presence or absence of that protein approximates a binary signal. If we have binary values, it is natural that we might want to perform simple logical operations upon them. This is not entirely straightforward. The problem lies in the fact that there is no well-defined concentration corresponding to '1' in protein logic. Any substantially non-zero concentration encodes '1', but the actual level depends on factors such as the decay constant, how recently the cell has divided, etc. It is therefore difficult to design, for example, an AND mechanism that will fire when both of its inputs are non-zero but not when just one of them is unusually high. (Note that devising an inclusive-OR mechanism is trivial.)

The key to solving the problem relies on the fact that there is a well defined '0' value, namely a concentration of 0.0. Thus, logical operations where all the clauses are negated can be carried out. It is straightforward to make a NOT mechanism by using a gene with a slightly negative bias, high steepness, and a large negative weighting on the input protein promoter. Hence, $P \wedge Q$ can be expressed as $\neg(\neg P \vee \neg Q)$. For example, when trying to pick out cells at a certain distance from a signalling cell (see below), we have found it useful to have a 'right distance' protein that is expressed when neither a 'too close' nor a 'too far' protein is present.

Location-specific cell selection. To build all but the simplest shapes, it is necessary to pick out cells to perform special functions according to their position in the embryo. A single cell can only produce a radially symmetric signal gradient, and hence any specific level of external protein is shared by a sphere of cell-sites. Uniquely pinpointing one location requires the intersection of four spheres, which requires four cells in different places emitting different signals. However, we have a chicken-and-egg problem: how can we position these cells, since they are required for positioning?

One solution we have found to be useful is as follows. Assuming one signal cell is already set up, the first cell which finds itself at the correct distance out can become the second signal cell, hence breaking the symmetry arbitrarily. Then, the first cell to find itself at appropriate distances from both existing signal cells can become the third, and so on. Of course, it is essential that once one cell has assumed a particular signalling role, none of the others do. This can be achieved if the presence of its signal inhibits other cells from emitting that signal. But this presents a problem: the concentration of the signal protein is obviously at its highest at the signalling cell. So how can we prevent it from inhibiting itself?

One method is to use an accumulator protein to create a delay (see above). Being at the correct distance from the existing signal cell(s) causes a certain protein to be produced. This protein triggers the production of the accumulator. Once the accumulator reaches a threshold level, an autocatalytic marker is produced, locking the cell into its role and causing the signal to be emitted. If, however, the signal is detected at any point during the waiting period, the process is immediately terminated. This mechanism ensures that only the first cell to end up in a suitable position will start signalling.

Once the multi-signal system is set up successfully, similar techniques can then be used for placing morphological features on the embryo. Returning to the cigar example, it would be possible to identify an outer cell and produce proteins which would lead to the production of *axis* and to the spindle being oriented in the direction of least *signal*. In this way, a 'limb' extending outwards from the central 'body' could be created (see the 'Quadruped' demo online).

In some cases, we may not want to globally inhibit other cells from responding when one does. For instance, we might want to set up two or more sites from which limbs will grow. In this situation we can alter the sensitivity to the signal such that only cells within a certain radius will be inhibited. Such lateral inhibition is useful for creating regularly-spaced structures, e.g. hairs on skin.

6 Summary

We have introduced and described METAMorph, a software platform for the experimental design of genomes encoded as genetic regulatory networks, given an example model of the development of a cigar-shaped body, and presented some of the design techniques that we have found useful. Although these are only a small step towards qualitatively characterising the expressive bias of the model of GRN used in METAMorph, we expect that further experimentation and open development will contribute to a more complete picture over time.

References

1. J.C. Bongard. Evolving modular genetic regulatory networks. In *(Proc. IEEE 2002 Congress on Evolutionary Computation (CEC2002)*, pages 1872–1877. IEEE Press, 2002.
2. P. Eggenberger. Cell interactions as a control tool of developmental processes for evolutionary robotics. In *From Animals to Animats 4: Proc. 4th International Conference on the Simulation of Adaptive Behavior*, Cambridge, MA, 1996. MIT Press.
3. P. Eggenberger. Evolving Morphologies of Simulated 3D Organisms Based on Differential Gene Expression. In P. Husbands and I. Harvey, editors, *Proc. 4th European Conference on Artificial Life (ECAL97)*, Cambridge, MA, 1997. MIT Press.
4. S. Kumar and P.J. Bentley. An Introduction to Computational Development. In S. Kumar and P.J. Bentley, editors, *On Growth, Form and Computers*, chapter 1, pages 1–44. Elsevier, 2003.
5. T. Quick, C.L. Nehaniv, K. Dautenhahn, and G. Roberts. Evolving embodied genetic regulatory network-driven control systems. In W. Banzhaf, T. Christaller, P. Dittrich, J.T. Kim, and J. Ziegler, editors, *Proc. 7th European Conference on Artificial Life (ECAL2003)*. Springer Verlag, 2003.
6. T. Reil. Dynamics of gene expression in an artificial genome – implications for biology and artificial ontogeny. In D. Floreano, J.-D. Nicoud, and F. Mondada, editors, *Proc. 5th European Conference on Artificial Life (ECAL99)*, 1999.
7. T. Taylor. A genetic regulatory network-inspired real-time controller for a group of underwater robots. In *Proc. 8th International Conference on Intelligent Autonomous Systems (IAS-8)*, March 2004.

Acknowledgements

Facilities for this research were provided by the Institute of Perception, Action and Behaviour in the School of Informatics, University of Edinburgh. This work forms part of the HYDRA project (EU grant IST 2001 33060, http://www.hydra-robot.com).