UNIVERSITY OF MANITOBA FACULTY OF ENGINEERING

# Morphogenesis Simulation

Matthew Woelk
May 27, 2013

## 1 INTRODUCTION

This report describes a project which simulates a simplified morphogenesis process through cell growth and cell differentiation. Agent based modeling, cellular automation, and message passing were each used in the development of this project.

This project succeeded in its three goals. The first was to have an organism which would start as a single cell which would then grow and divide until it became a multi-celled organism; the second was to have the organism develop a pattern; and the third was to have the organism take a form or shape which had some form of structure.

## 2 BACKGROUND

Morphogenesis is the process which "controls the organized spatial distribution of cells during the embryonic development of an organism"[1]. It occurs in the first few days of animals' lives, when they are growing from a single cell into a multi-cellular organism. This is an interesting process to study because of the interactions between cells which, though simple, produce animals which are very complex systems.

Inspiration for this project came from two areas. One, from the computational side of the project, was cellular automata. This motivation came from Stephen Wolfram's book "A New Kind Of Science"[2]. The other area of this project was the behaviour of cells during morphogenesis. This inspiration came from talks with Richard Gordon[3].

For this project I have assumed that cells are able to do the following things: For the process of pattern formation, cells must be able to

- divide into two cells of similar types
- divide into one cell of a similar type and one of a different type

and for the process of taking shape, cells must be able to

- detect the types of their neighbours
- send a signal to only one of its neighbours
- send a signal to one neighbour of a specific type
- change their type
- respond based on their surroundings

The mechanism by which real cells could succeed at these tasks is beyond the scope of this paper.

## 3 Implementation: Engine and Physics

This project was implemented in HTML, CSS, and Javascript using the d3.js[4] visualization library. Each cell's nucleus is a node of a d3.js force layout. There are no links between nodes, but they have friction, positive gravity which pulls them together, and a negative charge which pushes them apart.

Cells do not grow in size, but they can split into two equally-sized cells. There is inherit randomness in the locations of the cells, so no two simulations are identical. Cell walls are drawn using a voronoi diagram, clipped to circles.

A diagram of the software interface is shown in Figure 3.1.

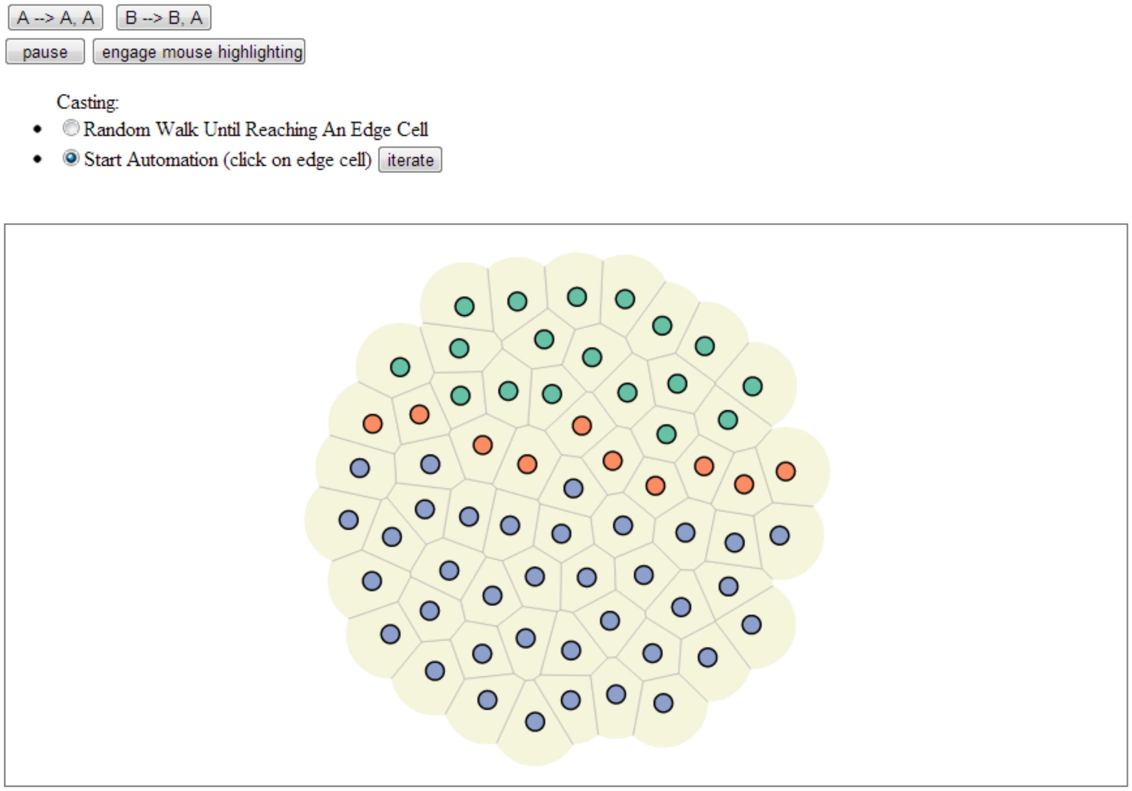
## 4 Implementation: Pattern Formation

Pattern formation was achieved by starting as one cell of a certain type, then splitting that cell using a sequence of cell differentiations. Two types of splitting were used: symmetric, where one cell of a type becomes two cells of that same type, and stem cell asymmetric, where one cell of a type becomes two cells: one of the same type and one of a second type. An example of these processes are shown in Figures 4.1, 4.2, 4.3, and 4.4.

For this project, sequences of six splits were used, and for each a different pattern was formed. The following are four example sequences. For simplicity, 3-AB will indicate three stem cell asymmetric splittings in a row, and 2-AA will indicate two symmetric splittings in a row.

The sequence 2-AA, 1-AB, 3-AA gave a large blotchy result, comparable to a cow's spots. This is shown in Figure 4.5. The sequence 4-AA, 1-AB, 1-AA gave a pattern of long streaks, comparable to a zebra's lines, though not as straight, as shown in Figure 4.7. The sequence 2-AA, 3-AB, 1-AA gave small dots. This is shown in Figure 4.6. Finally, the sequence 1-AB, 5-AA produced a pattern where one side's cells are of one type, and the other side is of another, as can be seen in Figure 4.8.

Therefore, through sequences of cell differentiation, I have shown that patterns can be formed using two types of cells, and two types of differentiation.



Casting:

- ○ Random Walk Until Reaching An Edge Cell
- ● Start Automation (click on edge cell) iterate

Figure 3.1: User interface for Morphogenesis Project

## 5 Implementation: Shape Formation

Two techniques were used to allow the simulated organism to take shape. The first comes from agent based modeling techniques, called message passing; and the second comes from cellular automata, and is the method of having cells choose their next state based on simple rules. It is an asynchronous cellular automaton because cells do not all calculate their next state before implementing them in unison, rather each one will calculate and change its state before another does. This is an implementation detail which I do not think strongly affects the outcome.

To grow a shape, first a single unique cell must exist among many other identical cells, which the head and spine will grow from. This method requires the original cell to have some object inside of it which is not replicated with the cell. Rather, there is only ever one of the object. Once enough cells have been formed, the organism can start taking shape. In this project, the first cell is stored at an index of zero in an array, which is how it is stored as being unique. This cell could end up anywhere in the organism at the time of shape formation. Any number of cells can be chosen as being enough for this stage, but 32 or 64 are recommended for visually interesting results.

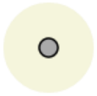


Figure 4.1: The starting cell.

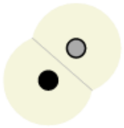


Figure 4.2: A stem cell asymmetric split.

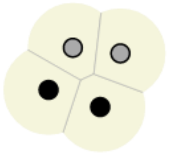


Figure 4.3: A symmetric split.

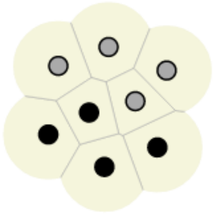


Figure 4.4: Another symmetric split.

For the spine-growing algorithm to work, the spine must be started at the edge of the organism. To find a unique edge cell, the single unique cell sends a message to one of its neighbours. Each neighbour who hears it in turn sends this message to one of its neighbours as shown in Figures 5.1 and 5.2. This continues until a cell which is on the boundary receives the message, at which

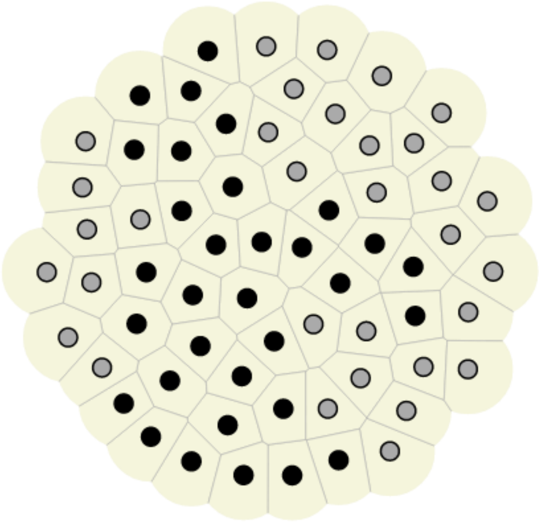


Figure 4.5: A blotchy pattern.

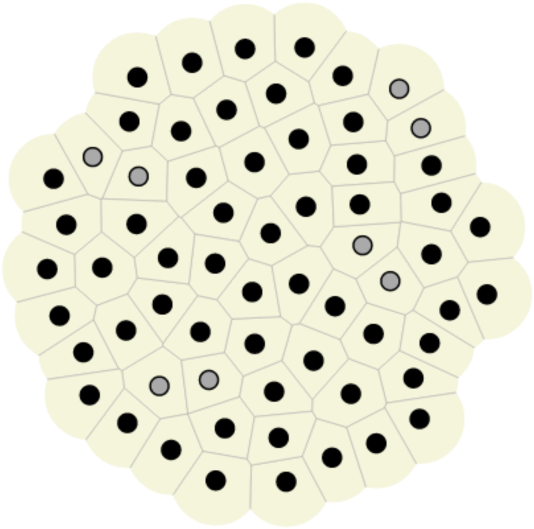


Figure 4.6: A dotted pattern.

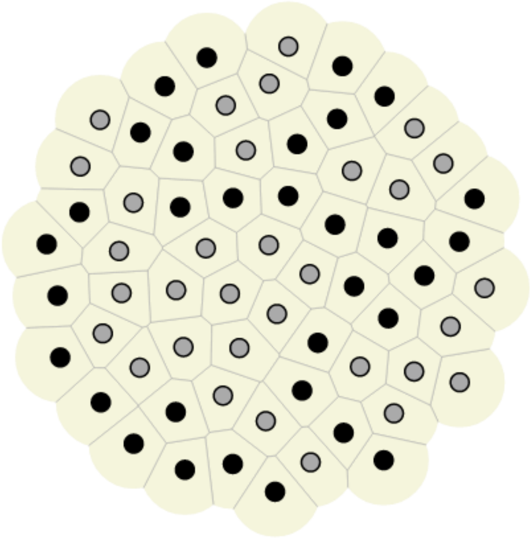


Figure 4.7: A streaky pattern.

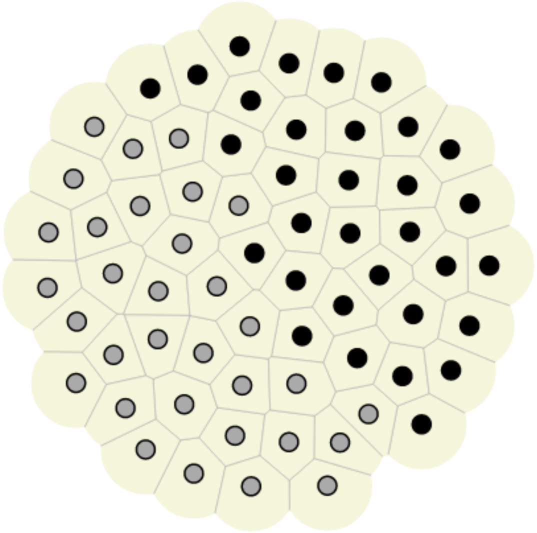


Figure 4.8: A half and half pattern.

point it begins the spine-growing algorithm. In the figures, the chosen unique edge cell has grown head cells to indicate that it was chosen.

The spine-growing algorithm works by signalling one cell beside it, which is also a boundary cell, to become the left side of the animal, and the other neighbouring boundary cell to become

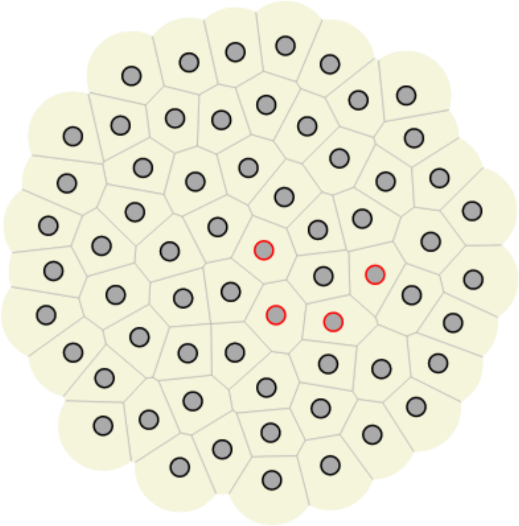


Figure 5.1: The beginning of the trail of messages being send from cell to cell.

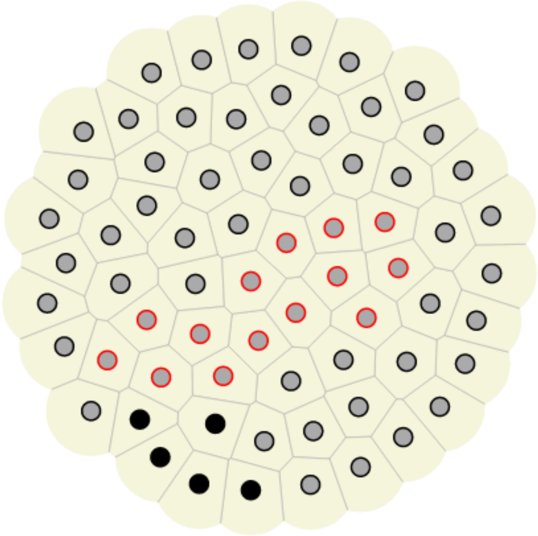


Figure 5.2: The conclustion of the trail, ending with head cells being grown.

the right side of the animal as shown in Figure 5.3. The furthest-left orange cell is the originator of the algorithm, and the two neighbouring edge cells, one of which is green and the other blueish, are the two signalled cells. Every cell in the organism follows three simple rules, which, once the left and right origin cells have been established, automatically form the left side, the right side, and the spine of the organism, as can be seen in Figures 5.3, 5.4, 5.5, and 5.6.

The three simple rules are as follows:

- Any cell which has both a "left" and a "right" neighbour will become a spine.
- Any cell which has a "left" neighbour, but no "right" neighbour will become a "left" cell.
- Any cell which has a "right" neighbour, but no "left" neighbour will become a "right" cell.

This has the result of growing a shape which has a spine roughly down the middle, along with two distinct sides, as can be seen in Figure 5.6.

## 6 Future Work

To improve similarities between simulated cells and real cells, there are a few improvements which could be made.

One possible improvement to this simulator would be to give the cells the ability to grow before they divide, and to have that process be automatic.

Another improvement would be to add structure to the cells. Currently, the cells all attract each other with a gravity, and repel with a negative charge, but cells could be made to cling to

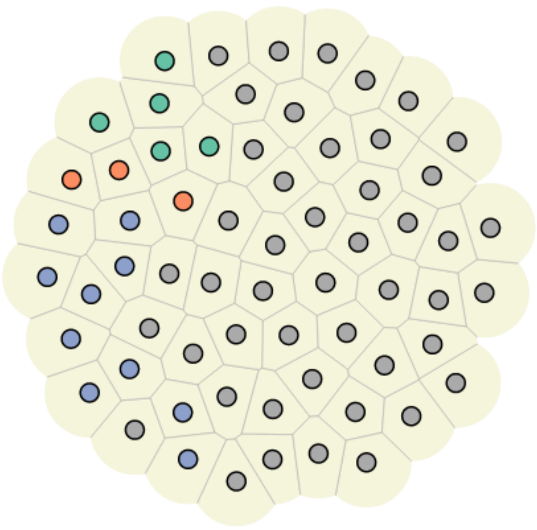


Figure 5.3: The first iteration of spine-forming automation.

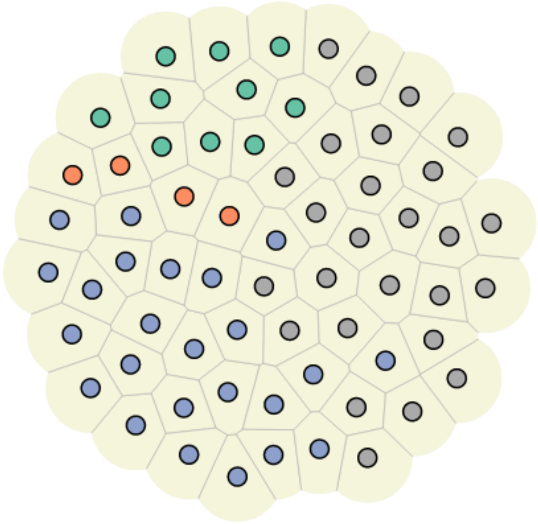


Figure 5.4: The second iteration of the automation.

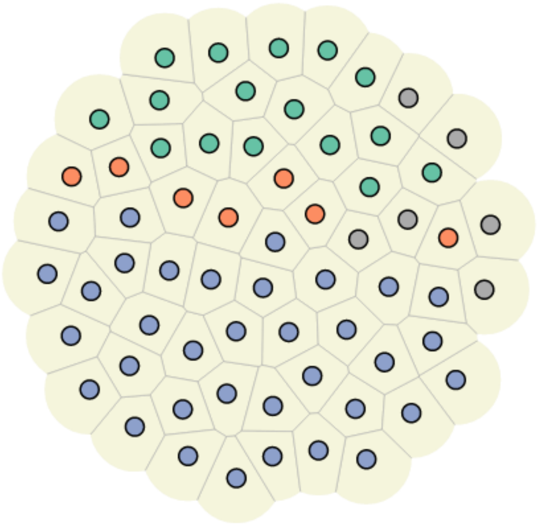


Figure 5.5: The third iteration of the automation.

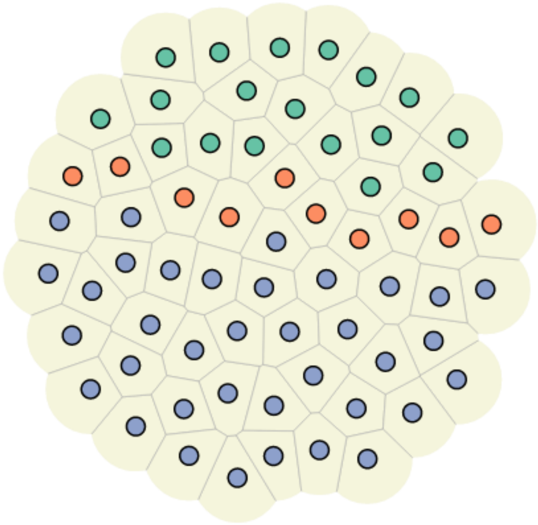


Figure 5.6: The final iteration of the automation.

each other. In addition, cell shapes could vary; instead of all cells approximating circles, they could vary depending on their type.

Further approximation of real cells would also require a three dimensional simulator. For this to succeed, the algorithms would have to be rethought.

Additional automation routines could be added to add life to the resulting animal. Blood circulation and walking are two examples of this.

These automations could then be triggered so that, for instance, an animal could grow, differentiate, produce a pattern and structure, and then start walking.

## 7 Conclusions

This project has demonstrated a simulator of cells in the early stages of animal growth. Techniques of agent based modelling, cellular automata, and message passing were used to simulate morphogenesis through cell division and differentiation. This project succeeded in its three goals of animal growth, pattern formation, and shape formation.

This project can be demoed at `http://www.mattwoelk.ca/ABM/morphogenesis.html` and the source code can be found at `https://github.com/mattwoelk/ABM_Morphogenesis`.

## References

[1] Wikipedia, Morphogenesis — Wikipedia, The Free Encyclopedia, 2013, `http://en.wikipedia.org/w/index.php?title=Morphogenesis&oldid=552613458`, [Online; accessed 23-May-2013]

[2] A new kind of science, Stephen Wolfram, 2002

[3] Richard Gordon, `http://embryogenesisexplained.com/`

[4] d3.js javascript library, `http://d3js.org`