# Pattern formation through minimalist biologically inspired cellular simulation

Marcelo de Gomensoro Malheiros*
UNIVATES, Brazil
Universidade Federal do Rio Grande do Sul, Brazil

Marcelo Walter†
Institute of Informatics
Universidade Federal do Rio Grande do Sul, Brazil

## Abstract

This paper describes a novel model for coupling continuous chemical diffusion and discrete cellular events inside a biologically inspired simulation environment. Our goal is to define and explore a minimalist set of features that are also expressive, enabling the creation of complex and plausible 2D patterns using just a few rules. By not being constrained into a static or regular grid, we show that many different phenomena can be simulated, such as traditional reaction-diffusion systems, cellular automata, and pigmentation patterns from living beings. In particular, we demonstrate that adding chemical saturation increases significantly the range of simulated patterns using reaction-diffusion, including patterns not possible before such as the leopard rosettes. Our results suggest a possible universal model that can integrate previous pattern formation approaches, providing new ground for experimentation, and realistic-looking textures for general use in Computer Graphics.

**Index Terms:** I.3.7 [Computer Graphics]: Three-Dimensional Graphics and Realism—Color, shading, shadowing, and texture I.3.8 [Computer Graphics]: Applications I.6.8 [Simulation and Modeling]: Types of Simulation—Discrete event

## 1 Introduction

Although much is now known about the genetic code and the biochemical mechanisms that occur inside living organisms, we still understand little about the actual processes that control growth, determine form, and define skin pigmentation [24]. The seminal work of Alan Turing provided a diverse, yet controversial direction. An abstract chemical reaction, based on a simple two-reagent system and modeled by differential equations, gave rise to the possibility that very simple mechanisms could provide biological diversity. Turing elegant speculations brought to light the concept of reaction-diffusion (RD) systems [25], which later led to several important studies on Biology [10, 17, 20] and Computer Graphics [7, 26, 30].

In parallel, the early work of Turing and von Neumann led to the theoretical basis of computing, where simple abstract machines were proven to have the computational power to run arbitrarily complex algorithms. Cellular automata (CA) were born [28], and led to the design of very simple yet powerful rule-based systems, like Game of Life [9] and L-systems [22]. In fact, Computer Graphics has been widely using RD and CA as base models to generate pigmentation patterns. Further pioneering work on forms [8], growth [6] and simulation [29] made modeling natural phenomena a diverse and multidisciplinary area. However, most of the techniques were still very specific to the fields they deal with, being sometimes *ad hoc* and quite complex.

In this paper, we seek to combine a few of the cited techniques into a minimalist yet biologically plausible cellular simulation and explore the process of diffusion-driven pattern formation. Such model should reproduce, yet in a simple form, already known and fundamental mechanisms of living cells. We reinforce that such plausibility is pursued both qualitatively (by replicating basic cell processes) and quantitatively (depending on large numbers of cells to produce an emergent pattern). Figure 1 shows a result of our model. We simulated the pattern for a particular moray eel (*Muraena melanotis*) using a *single* RD equation, where we set distinct diffusion rates at different parts of the skin, enabling a saturation limit for the two chemical reagents. The visual result is very close to the actual pattern as we can see the large distinctive black spot along the irregular white spots.

Figure 1: The moray eel *Muraena melanotis*: actual photo (above) and simulated pattern mapped onto a 3D model (below). Photo courtesy of Richard Bowes.

## 2 Related Work

There is extensive literature that discusses reaction-diffusion systems in different knowledge areas such as Mathematical Modeling, Physics or Theoretical Biology. In Computer Graphics it has been pursued as a general pattern generation mechanism in a few landmark studies, like [7, 26, 30]. More recently, a few specialized applications were presented [2, 13, 23]. In fact, Developmental Biology is doing most of the current research relating biological patterns and RD [15, 24].

On the other hand, there are many works on cellular automata and growth models which usually follow the general assumption that simple rules can produce complex results. Some pioneering

*e-mail: mgm@univates.br
†e-mail: marcelo.walter@inf.ufrgs.br

16-19 May, Edmonton, Alberta, Canada

works pursued growth models following a biological perspective, like [1, 22]. However, very few indeed combined both reaction-diffusion and growth rules within a cellular simulation.

Fleischer [6] proposed a general model inspired by biological mechanisms. There was a concerted effort to simulate not only the physical contact between cells but also their chemical and electrical communication, along with interaction with the surrounding environment and support for cell division. The authors opted to employ very general rules, in a high-level language reminiscent of C, which added a high complexity to the design of experiments. Interestingly, their model incorporated diffusing chemicals and the possibility of running a reaction-diffusion system, albeit restricted to a standard regular grid that both covered cells and the extracellular space. That is, the chemical concentrations were not tied to specific cells, but simulated as an environmental property.

While [26] was the first to simulate an RD system on the faces of a mesh, [4] has shown that the same simulation could run on unconstrained and free-moving cells. The context of the latter work was a more general computing model inspired by biological concepts. A recent work [27] employs a simulation for zebrafish skin patterns, where individual cells are modeled and interact to generate pigmentation. This model strongly depends on cell migration but otherwise have mechanics similar to our own. However, the focus is on the reproduction of particular biochemical phenomena that matches the real experiments made.

We have thus identified the opportunity for exploring biologically inspired designs that pairs at the same level RD systems and growth rules, balancing complexity and generality. This paper summarizes our findings so far.

## 3 Proposed Model

We propose a model that can account for several types of pigmentation patterns; it is both simple and biologically plausible. We have focused on a minimalistic approach, avoiding *ad hoc* solutions and keeping its design strictly adhering to chemical, mechanical and biological occurring mechanisms. In fact, we use simplicity as an Occam's razor to guide the development of a reduced set of features that can still provide a large diversity of patterns, being easily controlled. Our primary emphasis is on the *mechanochemical* interaction between simulated biological cells, mainly chemical diffusion through membranes and damped collision, besides controlled cell division.

### 3.1 Design rationale

We are particularly interested in pigmentation patterns which are mostly influenced by cell interaction on the organism skin. Thus our model works at a mesoscale level, abstracting the living cells as indivisible entities. Therefore, we opted to simulate cells as compact units with a homogeneous state. A set of rules defines the genetic programming of a cell, triggered only by simple conditions, and which results in just simple cell events. All simulated cells bear the same genetic code, or in other words, are bound by the same set of rules. Cells can differ regarding their internal state, which will then trigger different events at different times during the simulation blah.

Although most growth processes happen in 3D inside living organisms, our focus in pigmentations patterns justifies the constraint of having cells on a bidimensional domain, analogous to the surface of the skin. It can be argued that surface curvature can affect the creation of patterns, but we opted to have a simpler environment; curvature can still be eventually simulated by activating different rules based on localized concentrations of some reagent.

We devised the extracellular space as empty, with no cells but with plenty of space for growth. Hard limits could be imposed to the cell simulation if needed, constraining the growth to a fixed bounding box to represent cell compression. We have also opted for strict locality for cells during the simulation since it is a biological fact that most signaling happens when cells are touching their membranes. Therefore, by using local interactions the state change for a given cells depends only on its nearest neighbors. This assumption both simplifies implementation and provides more predictability, without loss of expressivity. Furthermore, such locality also makes possible the simultaneous and independent evaluation of cells, thus being amenable to parallel implementation on a GPU.

It directly follows from these choices that there is no global control or influence after the simulation starts. Cells only move when subject to a collision, caused by cell division. The reagent concentrations only change when subject to chemical diffusion, synthesis, or consumption. The only global mechanism that is explicitly allowed is the passage of time, in the form of activation of rules dependent on the iteration count since the simulation starts. We have explicitly not implemented individualized cell rules. Thus it is not possible to have a rule that affects only a particular cell. Nevertheless, such cell can be initialized with a different reagent concentration, which then triggers a specific rule during the simulation.

We justify our model based on the premise that although the biological pattern generation mechanisms in the literature are quite varied on their internal processes, many follow the outcome of the simpler conceptual model of "short-range activation and long-range inhibition" [17]. That is, even if the chemical and biological mechanisms are intrinsically different, the net result is very similar: the establishment of Turing patterns and stationary waves [15]. So it is valuable to employ a generic yet theoretical reaction-diffusion model as the basis for pattern generation in Computer Graphics, both for its simplicity and breadth of previous studies.

### 3.2 Cell state

A cell within the simulation is modeled as a homogeneous unit and geometrically as a unit circle. The numerical attributes for a single cell are summarized in Table 1.

Table 1: Attributes of a single cell.

| Attribute | Description |
|---|---|
| position | $x$ and $y$ coordinates |
| birth | iteration number the cell was born at |
| neighbors | number of nearby cells |
| polarity | unit vector, with $i$ and $j$ components |
| $R$ | concentration for reagent $R$ |
| $D_R$ | diffusion rate for reagent $R$ |

At the beginning of a simulation, each cell has their concentrations established and a set of chemical reagents defined. In analogy to the biological term, we call this initial configuration a *prepattern*.

Our model neither explicitly establishes different cell types nor discrete cell states. Thus any functional specialization by cells can only be the result of different reagent concentrations, which in effect can activate different rules. The actual meaning for any reagent is up to the design of the rules.

The cell membrane is thought as having varying porosity, thus being permeable or not to the diffusion of its reagents. The diffusion rate (or permeability) of each reagent is individualized and per cell. Each cell has an orientation, represented by a unit vector. Albeit cells are modeled as perfect circles, biological cells can assume *polarity* in the sense that they gain a preferred direction [11]. Polarity is a critical feature of our model, as we have a way to establish and maintain directionality during either division or diffusion. Cell polarity can be defined explicitly in the prepattern, as a direct result of cell division or by the activation of a rule that aligns the cell's polarity to the concentration gradient of a given chemical reagent.

Finally, each cell has a counter for how many neighbors it has. Neighbors are the cells whose membranes are touching, and which therefore can interact with this particular cell, either by diffusion or collision. This counter is updated at each simulation iteration and

gives a simple mechanism to evaluate whether a cell is under strong compression or whether it is in a tissue border. It is also used to prevent excessive division.

### 3.3 Diffusion and anisotropy

A fundamental idea of our model is to decouple reaction from diffusion. Diffusion is seen as a simple chemical process, where a given substance spreads within some substrate. Therefore, diffusion occurs continuously among cells, at every simulation step. On the other hand, reaction is seen as a per-cell condition-triggered event, and therefore must be explicitly initiated by some rule (Section 3.4).

Naturally, diffusion occurs through the cell membranes, so it is controlled by the diffusion rate of each cell involved in the process. As the actual diffusion is computed as a summation of pairwise cell contributions, the actual diffusion rate is a combination of the individual diffusion rates of the two cells in contact. If one of such cells is impermeable, no diffusion will occur between them, and therefore besides collision, the only possible interaction between nearby cells is chemical diffusion. Figure 2 illustrates normal diffusion between three cells.

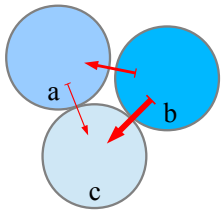


Figure 2: Normal diffusion: *b* has highest concentration and *c*, lowest. The amount of diffused reagent is indicated by the width of the red arrows.

Being empty, the extracellular space does not allow diffusion, which is equivalent to a zero-flux or Neumann boundary condition. Therefore, each group of nearly placed cells can be thought of a closed system, functioning independently from other connected groups of cells within the same simulation.

Diffusion is normally isotropic. However, we have also made possible *anisotropic diffusion* for a given chemical reagent. Such diffusion can be constrained to happen in the same direction given by the cell polarity. As such, the actual amount transferred between cells is modulated by their relative orientations, as shown in Figure 3. Anisotropy has been used to constrain reaction-diffusion systems in several previous works [14,23,30], but always within a fixed and regular cell arrangement.

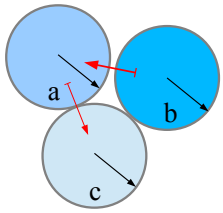


Figure 3: Anisotropic diffusion: cell polarity is indicated by the black arrows.

### 3.4 Cell events

Here we give an overview of the discrete cell events that can happen within the simulation. Any such event is triggered when a specific condition is met, as described by the set of rules that comprises the genome for a particular experiment. Each rule has a condition and an action. The conditions are restricted to testing for either the iteration counter or simple comparisons between current cell attributes and real values. When a condition is met, the action is immediately performed, which results in the occurrence of one of the cell events described in Table 2.

Table 2: Possible cell events within the simulation.

| Action | Cell event |
|---|---|
| change | produce or consume reagent *i* |
| change | change the diffusion rate for reagent *i* |
| react | modify concentration according to a RD equation |
| divide | perform instantaneous cell division |
| polarize | change polarization along concentration gradient |

The simplest event is the production or consumption of a chemical reagent within the cell. By design, an action cannot set the internal concentration of a reagent to a specific value. It is only possible to increase it or decrease it by a given amount. This was established based on two particular arguments. First, although we are simulating discrete cells within discrete time steps within the simulation, we are modeling real phenomena where concentrations change continuously. So it makes sense to either produce more or consume part of the chemical diluted inside the cell. Second, reagent concentration is changing constantly as the result of chemical diffusion, then such change is added to the production or consumption of a given reagent.

There is no restriction on the amount of reagent either produced or consumed in a single action. However, chemical concentration cannot be negative and can be bounded above by a certain saturation amount. Such saturation level is optional, being globally defined for each reagent, and cannot be changed during the simulation. Besides the change in a reagent concentration, actions can also change the current diffusion rate for a given reagent.

A critical part of our model is the reaction, which is the equivalent part of a reaction-diffusion system. When a rule activates reaction, two or more reagents inside the current cell are modified through the application of a specific set of RD equations. As said before, whereas diffusion happens continuously, the specific reaction part must be triggered.

Another event is cell division. When triggered, mitosis occur, and a new cell is instantaneously created. It is identical to its parent, except for two attributes: birth and polarity. Its birth is set to the next iteration, whereas the polarity for a child cell is defined relative to its parent's polarity, following an angle given as parameter for this action. A division event is usually tied to a probabilistic condition, where the chance of happening is explicitly set.

We have also defined a simple mechanism to prevent excessive division. Biological tissues have complex growth controls to regulate division and thus maintain its integrity. We have opted to employ a simple approximation for cell compression that prevents division above a certain limit: a global parameter can be used to prevent cells from dividing if they have more than a given number of close neighbors. It is interesting to note that no explicit mechanism for cell death is needed, as the combination of a uniform chance of mitosis and a maximum division limit seems to be enough so far to provide consistent growth of tissue-like structures.

Finally, we have a "polarize" event that orients a cell towards the higher concentration of a given chemical reagent. As diffusion is a process that happens for all chemical reagents, their concentrations are usually continuous. Therefore, we can derive a local chemical gradient by only examining the concentrations of the closest cells, providing a simple but powerful control mechanism, were many cells can be consistently oriented by setting up a single cell producing a chemical reagent. Such approach makes possible the exploration of several hypotheses related to the theory of morphogen gradients [24].

## 4 Implementation

We have set the specific requirement for the implementation to be scalable, in the sense that it must be able to handle a large number of cells. In fact, we pursue a directly parallelizable simulation loop, amenable to be run entirely on the GPU in the future. This demand reinforces several design choices already described before, particularly the strict locality of cells and the sharing of the same static set of rules during simulation.

The interaction between cells, either diffusion or collision, depends on locating all nearest neighbors within some influence range. Although this can be easily changed, we have settled for a radius of 3.0 centered at each cell. As each cell radius is 1.0, this enables the location of either the immediate eight neighbors in a typical Moore neighborhood (defined on a square lattice) or six neighbors (in a regular hexagonal lattice). In fact, most compact yet non-

regular arrangement of cells would result in an approximation for this hexagonal case. Figure 4 illustrates these three cases.

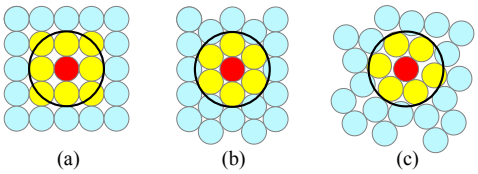


Figure 4: Neighborhoods: (a) square lattice, (b) hexagonal lattice, and (c) general situation. The black circumference marks the influence radius for the red cell. Cells within this range are shown in yellow.

Diffusion is performed by the change in the concentration of the current cell based on the sum of relative concentration differences among closest cells. The calculation follows Fick's second law, by using a numerical implementation of the Laplacian $\nabla$ operator over the nearest neighbors. In fact, the integration scheme is similar to a generalized five point stencil, where the center cell has the same weight as its neighbors. Eventually, the relative diffusion can also be modulated by the polarity of the current cell, having full diffusion with cells positioned over its main direction and gradually weaker diffusion along cells perpendicular to it. Currently, a Euler integrator is used, with a user-defined time step, as usually done in previous works [19, 23, 26]. Anisotropic diffusion is achieved modulating the full diffusion between two cells by using the dot product of their polarity vectors.

### 4.1 Nearest Neighbor Search

Our fundamental problem is the efficient location of nearest neighbors, a costly operation that has to be executed for each cell, during each simulation iteration. We have thoroughly researched the Nearest Neighbor Search (NNS) problem and refined the specific and little-known technique of *spatial sorting* [5]. In short, we have sought for a dynamic data structure that would not need to be rebuilt from scratch for each simulation step. We have evaluated that spatial sorting is optimal for our particular case, being memory efficient and capable of taking advantage of small local changes in the cell positions. Furthermore, by not relying on global counters or linked data structures, spatial sorting can be easily parallelizable. The major limitation is being an approximate NNS method, so care must be taken for the trade-off between precision and performance.

Therefore, besides spatial sorting, we have added a fast and specialized $k$-d tree as an alternative and exact NNS, based on Nanoflann, which can be used by the simulation. Moreover, we have also added a simple and very fast square lattice NNS, for static experiments where cells lay in a regular grid. In Section 5.7 we briefly discuss performance.

### 4.2 Code organization and interface

The implementation is called Pattern Explorer, organized into several modules. The more important modules are the NNS algorithms and the simulation loop itself. These are the ones that are planned to be rewritten in CUDA for running entirely on a GPU so that the rules can be loaded into the constant memory and cells then would be simulated in parallel. The simulation loop itself is quite compact, being about 500 lines of C++ code.

The simulation can be called from an API, thus text-mode standalone programs can be compiled. In fact, all timing calculations were done with this setup. We have also implemented a simple graphical user interface (GUI), using the OpenGL, FreeGLUT and AntTweakBar libraries. The GUI allows real-time visualization of the simulation, thus being able to monitor the pattern formation process and track the internal state of single cells.

To enable a fast cycle of experimenting different ideas for pattern formation, we have defined a simple text-based language, for specifying both the prepattern and the rules to be followed. Each text file is usually self-contained, which we call an *experiment*. The Supplementary Material gives a brief overview of this language, which is the same used to generate the results of this paper.

We also have other modules for output in different formats. We have an automated screenshot feature for capturing the evolution of patterns, an export for the cell positions and colors in vector format, and a high-quality image interpolation. This interpolation makes possible the generation of high-resolution textures, that can be applied to the 3D models. The interpolation algorithm itself is based on Natural Neighbor Coordinates, as provided by the CGAL 4.9 package. Source code for the complete implementation and all experiment files will be made publicly available[1].

### 4.3 Division and collision

We suppose that cells live within a limited environment. Since they are part of a tissue or immersed into a viscous extracellular fluid, their movement is highly constrained. Therefore, cells only move as a result of collisions, and collisions only happen after a division.

We have evaluated several possible mechanisms to model division, but have settled for the simplest one: division happens instantaneously, by creating a new cell at some small distance apart from the parent. Right after division, there is always some overlapping of the newborn cell with its parent and other nearby ones, which then causes collisions. Figure 5 illustrates the process.

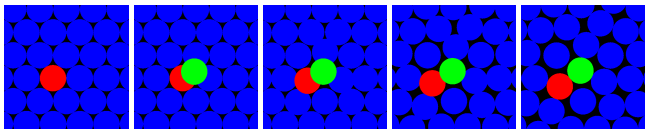

Figure 5: The red cell divides and the green cell is created. In the following iterations collision occurs and spreads all cells apart.

Collision testing runs for the same set of closest neighbors as diffusion, given by the fixed influence radius. When cells overlap we employ an impulse offset that makes them move apart. This scheme is a simplified version based on [3], as we do not need to explicitly compute resulting forces or maintain momentum for the cells. The actual offset is computed from the summation of impulses given by nearby cells and is proportional to how much overlap there is. The overall effect is that cells slowly push others apart, taking possibly several iterations until they are not overlapping anymore.

## 5 Results

Here we give an overview of relevant results obtained. Our model has a broad expressivity, as will be clear from the experiments, and its design allows ease of experimentation. The Supplementary Material presents more results.

We have chosen a slight variation of Turing's original RD system (as implemented by Turk in [26]) due to its simplicity. This model is given by Equations 1 and 2, where $\alpha$ and $\beta$ are usually fixed parameters. A *scale* parameter is given by $s$, which affects the intensity of the reaction part for both equations, having as effect the direct control of pattern wavelength. The chemical concentrations are given by $U$ and $V$, and their respective diffusion rates are $D_U$ and $D_V$. The actual diffusion is given by the Laplacian $\nabla^2$ operator.

In all examples shown here, we only used this RD model. Moreover, we have also fixed $\alpha = 16$ and $beta = 12$. We were interested in finding out how much variety we could achieve by adjusting only the scale factor (that only results in either chemical production or

[1] http://github.com/mgmalheiros/pattern-explorer

consumption) and the diffusion rates (which are just per-cell membrane permeability coefficients). That is, we have not varied the overall behavior of the RD model (which would happen by changes in *alpha* or *beta*), but restricted ourselves to "play" with naturally varying biochemical quantities.

$$\frac{\partial U}{\partial t} = s(\alpha - UV) + D_U \nabla^2 U \quad (1)$$

$$\frac{\partial V}{\partial t} = s(UV - V - \beta) + D_V \nabla^2 V \quad (2)$$

Figure 6 shows a *parameter map* using the Ready [12] package, that further mapped the kind of patterns we would like to reproduce. For our experiments, we further set as reference $s = 0.01$ and $D_V = 0.01$, and let $D_U$ vary within the interval $[0.0, 0.8]$. Then we can go from negative spots to labyrinthine patterns to larger spots, in relation to chemical $V$.

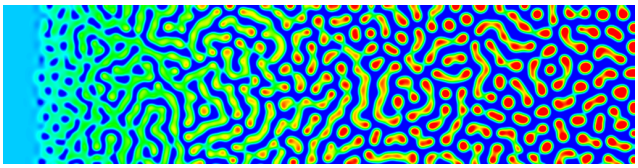

Figure 6: Parameter map for the Turing equations, ranging from blue (lower value) to red (highest) concentrations for chemical $V$. Diffusion rate for $U$ varies from 0.0 (on the left) to 0.8 (on the right).

Each experiment consists of three parts: global settings, the prepattern definition, and the rules to be run during the simulation. The input files for all results are available in the Supplementary Material.

### 5.1 Gradients and anisotropy

A simple, steady chemical gradient can be defined by two chemicals, $P$, and $G$. The first is used to mark producer cells, whereas the second defines the concentration gradient itself. Marked cells continuously produce some amount of G, whereas all other cells consume it. By adjusting the production and consumption rates, we can define the reach of the gradient. The resulting concentrations, mapped to a rainbow color map, are shown in Figure 7.

```
define chemical P // producer
define chemical G // gradient
use chemical P conc 0 diff 0
use chemical G conc 0 diff 0.1
create sqr_grid 20 2
set cell  0 chemical P conc 1
set cell 20 chemical P conc 1
rule if P conc == 1 change G conc  1
rule if P conc == 0 change G conc -0.06
```

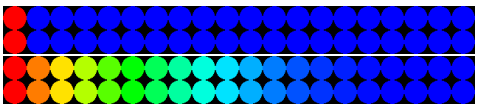

Figure 7: Simple stable linear gradient, for chemical $P$ (above) and $G$ (below). The experiment file is shown at top.

We may use a circular gradient, at the center of the pattern, to induce cell polarization. If we perform anisotropic diffusion, we can generate oriented stripes in a typical reaction-diffusion system. As already noted in [14], applying anisotropy to both $U$ and $V$ does not generate significative changes over the final pattern, but when applying to only one reagent, the results are striking. Figure 8 shows the result of enabling anisotropic diffusion only for $V$, where cells are oriented along a radial gradient: stripes orient along the radial lines. On the right side, we show the equivalent result from [23], which employed actual modulation of the RD equation parameters.

If we set anisotropy only for the $U$ reagent, the RD system now creates concentric, yet slowing moving, stripes, as seen in Figure 9. Note that the Turing parameters were only slightly altered from the previous experiment, yet we can achieve a very similar result to the one from [23], which still used manual modulation of parameters.

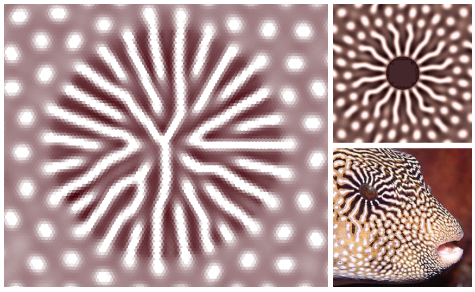

Figure 8: Effect of anisotropic diffusion for $V$ reagent only (left) and similar results from [23], made by parameter modulation (right).

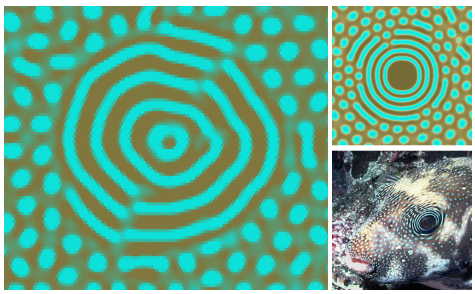

Figure 9: Effect of anisotropic diffusion for $U$ reagent only (left) and similar results from [23], made by parameter modulation (right).

### 5.2 Simple growth pattern

We can reproduce the pigment of a mollusk shell, like in [7], by creating a growing front of cells. On the left of Figure 10, we keep the row of cells at bottom both dividing and with active RD. After division, the pattern is frozen on the parent cells by making the diffusion rate drop to zero, whereas RD continues on the newborn cells, which again divide in the next iteration.

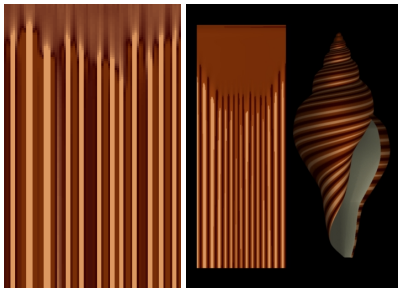

Figure 10: Growing front that freezes the older generated pattern (left) and result from [7] (right).

### 5.3 Pattern enlargement

A well-known feature of standard RD systems is that the pattern can adjust itself to changes in its domain, thus keeping an overall

wavelength for its features, like the distribution of spots or width of stripes. We studied a particular problem: after the hypothetical production of melanin-like pigmentation defines a pattern, how does it maintain its form during natural growth?

Suppose a uniform growth, where divisions occur with equal probability for each cell. If RD stays active during the process, after many cell divisions, the pattern would adjust to a larger area, and thus new details with the same wavelength would be introduced, as show in the (a) transition of Figure 11. However, if we stop diffusion to keep the concentrations, as in (b), division of cells would introduce exact duplicates, which then would add irregularities to the original pattern. On the other hand, if we only stop the reaction, diffusion would make the pattern fade and result in less-defined boundaries.

We have experimented with several mechanisms and settled for a particular one that is both simple and seems to be robust to global or local tissue growth. We "copy" the initial pattern from either $U$ or $V$ to another reagent, here called $P$, with a low diffusion rate (this is shown as the (c) transition in Figure 11). Then we add two mutually exclusive rules to reinforce $P$ concentrations close to either 0.0 or 2.0. When the current concentration is under 0.9, the first rule would gradually decrease P, whereas when the concentrations are above 1.1, it would increase $P$ towards the upper limit. The result is that these rules oppose the diffusion effect, by keeping the high and low concentration regions mostly unchanged during growth. The net effect is that some minor detail is lost, but the overall pattern is maintained, as in (d).

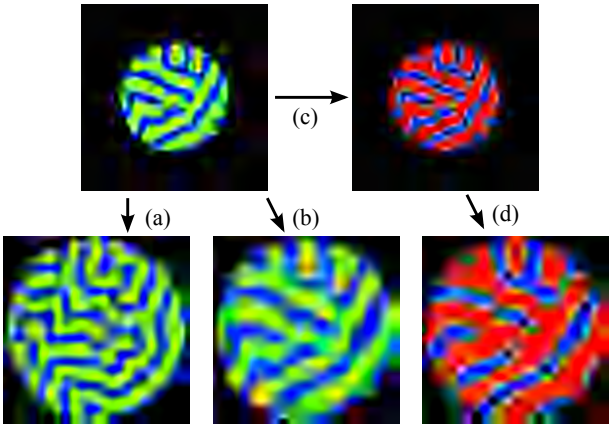


Figure 11: Distinct behavior during tissue growth. Transition (a) shows the effect of domain growth, whereas (b) shows cell division with inactive RD. The copy of concentrations to a third chemical (c) makes possible to keep the overall pattern under uniform growth (d).

### 5.4 Reinforcement mechanisms

Early versions of our model did not incorporate chemical saturation for the reagents. Some RD systems constrain chemical concentrations to be non-negative, and that was our only assumption from the beginning, as it makes sense in a real chemical process. However, we have since explored rules like the ones used for pattern enlargement and saw that imposing a limit concentration is both plausible and a common mechanism in biological organisms.

In fact, the explicit setting of an upper bound for chemical concentrations made possible a new category of patterns, based on a single reagent, which we called reinforcement patterns. We are not aware of previous studies discussing this mechanism. The three experiments shown in Figure 12 start with random concentrations for the single chemical $P$ and achieve stability under 2000 iterations. The resulting concentration for $P$ was then simply mapped to an interpolated color map.

```
define chemical P limit 1
use chemical P conc 0.5 dev 0.5 diff 0.01
create sqr_grid 100 100
rule if P conc > 0.7 change P conc +0.01
rule if P conc < 0.3 change P conc -0.01
```

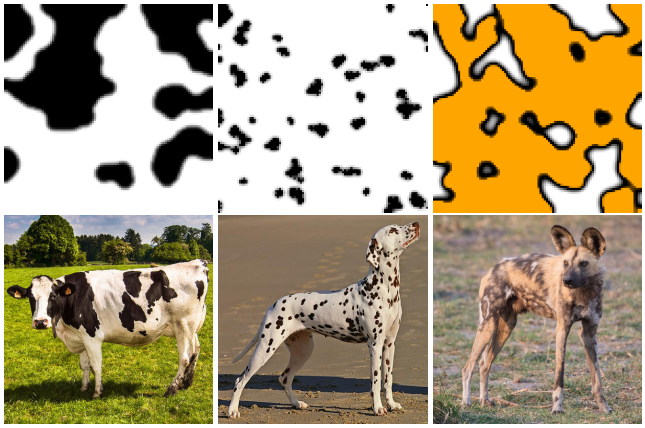

Figure 12: Stable reinforcement patterns: cow (*Bos taurus*, left column), Dalmatian (*Canis lupus familiaris*, center) and African wild dog (*Lycaon pictus*, right). The experiment file for the cow-like spots is shown on top. Photos in the public domain.

### 5.5 Saturated reaction-diffusion

We have discovered that the classic Turing RD system gains very compelling behavior when saturation limits are imposed to one or more reagents. The first example shows the emergence of a stable giraffe pattern from a random amount of initial concentrations for $U$ and $V$ chemicals, as in Figure 13. The RD system first produces irregularly spaced spots with similar size. Gradually, some spots increase and absorb smaller spots, creating larger regions, moving to a stable configuration.

```
define chemical U limit 6.3
define chemical V limit 6.3
use chemical U conc 4 dev 3 diff 0.04
use chemical V conc 4 dev 3 diff 0.01
create sqr_grid 100 100 wrap
rule always react U V scale 0.005 turing alpha 16 beta 12
```

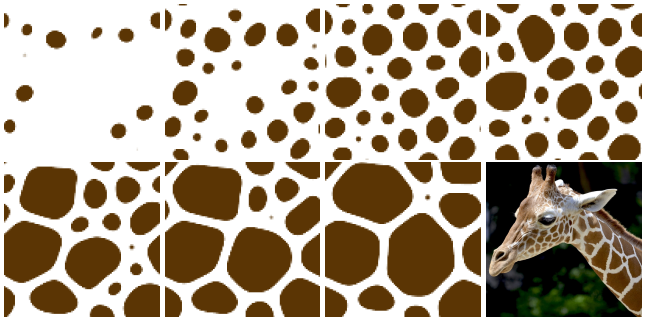

Figure 13: Reaction-diffusion where both $U$ and $V$ reagents achieve saturation, with emergent giraffe (*Giraffa reticulata*) pattern. The experiment file is show on top. Photo in the public domain.

Particularly striking are the dynamics provided by saturation: when one reagent hits its upper concentration limit, the other one is usually forced to zero. The use of saturation tends to create large areas of zero concentration and others of maximum concentration. This behavior is exactly what we were previously trying to achieve for pattern growing. That means that saturated RD is resilient to

domain growth, and now makes possible the creation of large areas, thus breaking the fixed wavelength from the standard Turing equations. Moreover, the implementation of saturation is trivial, being just a *max(limit, concentration)* operation. Finally, it is biologically plausible, being a universal chemical phenomenon.

When exploring the dynamics of the patterns generated by saturated reaction-diffusion, we found that the most visually interesting and capable of generating realistic biological patterns arrived from imposing a limit to either $U$ or the same limit to both $U$ and $V$. Figure 14 shows the parameter map for the situation where only $U$ is limited. Figure 16 shows a few randomly generated patterns.

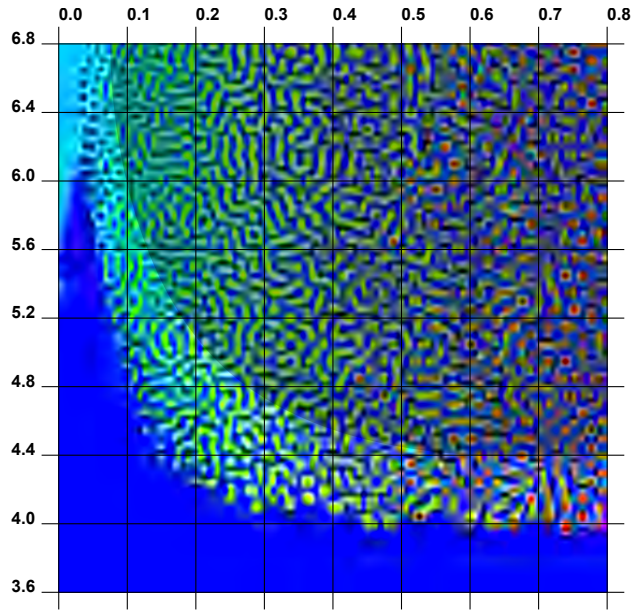


Figure 14: Parameter map for $V$, where $D_U$ varies the horizontal axis, and saturation limit $L$ varies in the vertical axis. In the darkened area the concentrations fluctuate below the limit, so there is no difference from the standard RD behavior. No steady patterns appear in the blue areas, as the reagents tend to constant concentration.

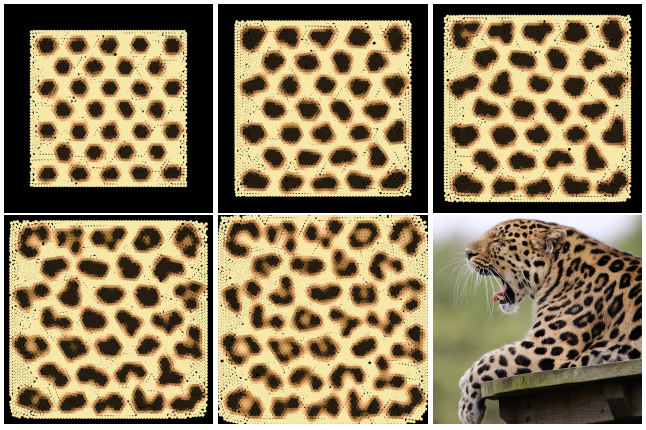

Figure 15: Emerging leopard (*Panthera pardus*) rosettes from combined uniform growth and saturated RD. Photo in the public domain.

We have confirmed the conjecture from [16] that leopard rosettes would spontaneously appear from the growth of the tissue, albeit with a simple mechanism and more accurate results, as shown in Figure 15. We have coupled uniform growth and saturated RD, imposing a limit only to $U$. To make the growth rate consistent over the simulated tissue, we further constrained the pattern into a growing square domain, and explicitly seeded the prepattern with uniformly spaced spots. Note that previous results for both giraffe and rosettes were only possible with cascade RD processes or more specialized computations [16, 26, 29, 30].

Figure 1 shows another experiment, where a simulation is run to generate the texture for a specific moray eel species, *Muraena melanotis*. We again employed saturated RD, involving $U$ and $V$. For the distinct parts of the body, we simply set different diffusion rates for $U$. The closeness of cells ensures the continuity of concentrations in the interface of such parts. For the distinctive black spot, we created an approximate elliptical gradient, based on two producer cells, and let nearby cells affected by this gradient to continually produce $V$, which prevented the emergence of white spots.

### 5.6 Cellular automata

Although our focus is on patterns created by RD systems, our model is also capable of running several types of cellular automata. In fact, we understand that it falls in the category of *continuous automata*, where valid states are no longer limited to be discrete numbers. Albeit there is no explicit support in our model for identifying who are the neighbors for a given cell (only the number of neighbors is available), we can control and measure local chemical diffusion as a mechanism to evaluate neighborhoods. In the Supplementary Material we present experiments implementing the classic Game of Life and Wireworld cellular automata, and also a more complex example of approximate Laplacian Growth.

### 5.7 Performance

Simulation time is dependent on five major factors: number of iterations, number of cells, number of chemicals, choice of rules and choice of NNS algorithm. That is, for each iteration and each cell we must evaluate all rules and then locate the cell's neighbors. If we consider a static regular grid arrangement, with cells having a fixed number of neighbors which can be located at constant time, we have an expected time complexity of $O(mn(r+c))$, being $m$, $n$, $r$, and $c$ the number of iterations, cells, rules, and chemicals respectively.

If growth is enabled, a regular grid can no longer be used, and we must either use an exact NNS, like a $k$-d tree, or an approximate one like spatial sorting. Then we must take into account that these NNS techniques need two distinct operations: overall setup and per-cell query. During setup, we must either build the $k$-tree from scratch or spatially sort cell positions, which are both done at each iteration. Then, for each cell, we have to perform a query to locate the actual neighbors. We can suppose an upper limit for the number of nearest cells because in practice we employ a limited influence radius and an explicit control to prevent excessive cell division.

The worst case time complexity for the $k$-d tree setup is $O(nlogn)$, whereas a single query operation is $O(logn)$. The overall complexity of the whole simulation using $k$-d tree is $O(mn(logn+r+c))$. We have analyzed spatial sorting in [5] and found an experimental complexity of $O(nlogn)$ for sorting, whereas the query can be done in constant $O(1)$ time. Thus, when using spatial sorting, we estimate the worst case time complexity also as $O(mn(logn+r+c))$.

Detailed timing information is given for all results in the Supplementary Material. We briefly discuss here two cases. The moray example has 20,000 cells, runs for 5,000 iterations and uses a square grid NNS, just taking 21 seconds to execute. The leopard example reaches 6,228 cells in 14,000 iterations. When run using a $k$-d tree, it takes 43 seconds, whereas with spatial sorting it takes 22 seconds. Spatial sorting is consistently faster than the $k$-d tree, and in this particular case, the average miss rate when detecting neighbors is 0.5%, using a 48 Moore neighborhood. The test machine is powered by an Intel Core i7-4500U CPU running at 1.80 GHz.

## 6 Conclusion

We have presented a parsimonious yet elegant simulation model, trying to mimic the fundamental workings of growing organisms, where the complexity of a few realistic biological patterns emerged from the combination of reaction, diffusion, chemical saturation and the application of simple rules.

We have strived to maintain minimalism and follow biological plausibility as our primary design goal. The resulting limited set of features made possible the systematic exploration of several combinations of RD parameters, timed and chemical conditions, and cellular actions. The results show that our model has the potential to generate patterns across a wide range of possibilities.

As future work, we plan to further evaluate the interrelation between growth and definition of cell polarity, as a mechanism for the emergence of oriented patterns. We also plan to implement the simulation loop entirely on GPUs, to simulate a larger number of cells. Furthermore, we plan to design an approach that makes possible the interactive and artist-oriented design of patterns.

Although we have only implemented Turing equations, it is straightforward to add Gray-Scott [21], Miyazawa's [19] or one of Meinhardt's equations [17, 18] to our model. Therefore, another future work is the exploration of the effect of chemical saturation on other, more complex RD models.

Figure 16: "Saturated Bunnies", made from randomly generated experiments.


## Acknowledgments

The first author thanks the institutional support from UNIVATES. The second author gratefully acknowledges the partial financial support from CAPES through grant BEX 5824/15-0.


## References


[1] P. Agarwal. The cell programming language. *Artificial Life*, 2(1):37–77, 1994.

[2] R. Barros and M. Walter. Synthesis of human skin pigmentation disorders. In *Computer Graphics Forum*. Wiley Online Library, 2016.

[3] S. Clavet, P. Beaudoin, and P. Poulin. Particle-based viscoelastic fluid simulation. In *Proceedings of the 2005 ACM SIGGRAPH/Eurographics symposium on Computer animation*, pages 219–228. ACM, 2005.

[4] D. Coore and R. Nagpal. Implementing reaction-diffusion on an amorphous computer. In *Proceedings of 1998 MIT Student Workshop on High-Performance Computing in Science and Engineering*, pages 189–192, 1998.

[5] M. de Gomensoro Malheiros and M. Walter. Spatial sorting: An efficient strategy for approximate nearest neighbor searching. *Computers & Graphics*, 57:112 – 126, 2016.

[6] K. W. Fleischer, D. H. Laidlaw, B. L. Currin, and A. H. Barr. Cellular texture generation. In *Proceedings of the 22nd annual conference on Computer graphics and interactive techniques*, pages 239–248. ACM, 1995.

[7] D. R. Fowler, H. Meinhardt, and P. Prusinkiewicz. Modeling seashells. *Comp. Graphics*, 26(2):379–387, 1992.

[8] D. R. Fowler, P. Prusinkiewicz, and J. Battjes. A collision-based model of spiral phyllotaxis. In *ACM SIGGRAPH Computer Graphics*, volume 26, pages 361–368. ACM, 1992.

[9] M. Gardner. The fantastic combinations of john conway's new solitaire game life. *Scientific American*, 223(10):120–123, Oct. 1970.

[10] A. Gierer and H. Meinhardt. A theory of biological pattern formation. *Kybernetik*, 12(1):30–39, 1972.

[11] N. W. Goehring and S. W. Grill. Cell polarity: mechanochemical patterning. *Trends in cell biology*, 23(2):72–80, 2013.

[12] T. Hutton, R. Munafo, A. Trevorrow, T. Rokicki, and D. Wills. Ready, a cross-platform implementation of various reaction-diffusion systems.

[13] J. T. Kider, S. Raja, and N. I. Badler. Fruit senescence and decay simulation. In *Computer Graphics Forum*, volume 30, pages 257–266. Wiley Online Library, 2011.

[14] T. Kim and M. Lin. Stable advection-reaction-diffusion with arbitrary anisotropy. *Computer Animation and Virtual Worlds*, 18(4-5):329–338, 2007.

[15] S. Kondo and T. Miura. Reaction-Diffusion Model as a Framework for Understanding Biological Pattern Formation. *Science*, 329(5999):1616–1620, Sept. 2010.

[16] R. Liu, S. Liaw, and P. Maini. Two-stage turing model for generating pigment patterns on the leopard and the jaguar. *Physical review E*, 74(1):011914, 2006.

[17] H. Meinhardt. *Models of Biological Pattern Formation*. Academic Press, New York, 1982.

[18] H. Meinhardt. *The algorithmic beauty of sea shells*. Springer Science & Business Media, 2009.

[19] S. Miyazawa, M. Okamoto, and S. Kondo. Blending of animal colour patterns by hybridization. *Nature communications*, 1:66, 2010.

[20] J. D. Murray. *Mathematical Biology. II Spatial Models and Biomedical Applications {Interdisciplinary Applied Mathematics V. 18}*. Springer-Verlag, 2003.

[21] J. E. Pearson. Complex patterns in a simple system. *Science*, 261(5118):189–192, 1993.

[22] P. Prusinkiewicz, M. James, and R. Mech. Synthetic topiary. In *Proceedings of the 21st annual conference on Computer graphics and interactive techniques*, pages 351–358. ACM, 1994.

[23] A. R. Sanderson, R. M. Kirby, C. R. Johnson, and L. Yang. Advanced reaction-diffusion models for texture synthesis. *Journal of Graphics, GPU, and Game Tools*, 11(3):47–71, 2006.

[24] A. Stathopoulos and D. Iber. Studies of morphogens: keep calm and carry on. *Development*, 140(20):4119–4124, 2013.

[25] A. M. Turing. The chemical basis of morphogenesis. *Phil. Trans. Roy. Soc. London*, B(237):37–72, 1952.

[26] G. Turk. Generating textures on arbitrary surfaces using reaction-diffusion. In *SIGGRAPH '91 Proceedings*, pages 289–298, 1991.

[27] A. Volkening and B. Sandstede. Modelling stripe formation in zebrafish: an agent-based approach. *Journal of the Royal Society Interface*, 12(112):20150812, 2015.

[28] J. von Neumann. The general and logical theory of automata. *Cerebral mechanisms in behavior*, 1(41):1–2, 1951.

[29] M. Walter, A. Fournier, and D. Menevaux. Integrating shape and pattern in mammalian models. In *SIGGRAPH 2001, Computer Graphics Proceedings*, pages 317–326, 2001.

[30] A. Witkin and M. Kass. Reaction-diffusion textures. *ACM Siggraph Computer Graphics*, 25(4):299–308, 1991.