Can coupling morphogenesis to a simple fate-specification model generate a zoology of lattice structures?

Kaile Yuan

March 11, 2020

Contents

1 Introduction **3**

1.1 Theoretical underpinnings of the phenomenon 6

reaction-diffusion system 6

Delta Notch intercellular signaling 7

(Corson et al. 2017) signaling driven model 9

vertex mechanical model of epithelial morphogenesis . 11

can coupling morphogenesis to a simple fate-specification model generate a zoology of lattice structures? 13

1.2 Computer models 15

SVG 16

asynchronous function 18

the Observable notebook 19

modeling environments previously employed 20

2 Methods **21**

random seeds and the maximum arbitrariness of meshes 21

class of objects 22

periodic boundary condition 25

3 Results and Discussions **27**

qualitative interpretation of the three parameters involved 27

two classes of cells 28

nonlinear mapping from u to K 32

4 Prospect **34**

morphological measure of cells 34

global shear motion might facilitate the regular patterning 34

1 Introduction

The dynamic behavior of epithelial cell sheets plays a central role during morphogenesis, and the basic players of it could be deduced to cells, a patch of cells, and subcellular metabolite networks given the problem of interest. Here we're interested in the intercellular mechanical aspect of epithelial cells, and opt to consult computational simulations when direct measurements of mechanical parameters are not accessible.

One interesting phenomenon in studying the epithelial cells is referred to as lateral inhibition(LI). Back in 1940, Wigglesworth observed cellular plaques in which each exerts an inhibitory influence around itself which prevents the development of new plagues. Later it's known to people that neurogenic genes *Delta*(Dl) and *Notch*(N), are one pair of instance underpinning the plague phenomenon. The signaling between Dl and N constitutes a prohibitive function in relation to the stable adoption of particular cell fates. (Skeath and Carroll 1992) has reported that the underlying family of proteins, helix-loop-helix(HLH) proteins, whose transient expression correlates with the resolution of proneural equivalence group. Equivalence group was coined to suggest the similar functionalities of cellular group during lateral inhibition patterning. (Artavanis-Tsakonas 1988) named a group of genes involved in neurogenesis as toporhythemic genes, reflecting the topology of such pattern.

(Goodyear and Richardson 1997) has studied lateral inhibition in chick basilar papilla, which is the auditory sensory organ of lizards, amphibians and birds, whose analogous organ in mammals being the organ of Corti. The oval-shaped organ measured 200 um - 1mm along proximal distal axis and 200 um along the superior inferior axis / medial lateral axis during the embryonic development, accompanied by the mosaic patterns varying between hair and supporting cells. Hair cells inhibit each other, thus during the development it's seen the emerge of hair cells along the proximal distal axis, interspaced by supporting cells. Hair cells start to emerge after embryonic day(E) 6, and till E12 a sigmoidal surge in amount is observed in the organ . The morphological determination, hinted by the epithelial lattice configuration difference, starts to take effect at E7. They has examined three regions to investigate such difference, central-distal(CD), superior-proximal(SP) and inferior-proximal(IP). One good indicator of topological difference is the hair and supporting cell contact number. At embryonic day 12, in terms of contact number IP has shown the most deviation from the E9, that on average 9.33(+-0.09)(n=24) supporting cells surround each hair cell, whereas in day 9 is 5.27(+-0.04)(n=28). Note that for neighboring number, even the least deviation from day 9 has exhibited statistical difference. In comparison, at E12 the contact number in SP and CD is 6.03 and 5.23 respectively. Moreover, if confine the scope to each equivalence group (IP, CD and SP), they found that the pattern is regular, which is also suggested by the comparatively small give and take value suffixed. One testable empirical formula for the reported contact number is the Lewis law(Rivier and Lissowski 1982), put forward in 1928 in Lewis' study of several two-dimensional cellular mosaics(cucumber epidermis, pigmented epithelium of the retina, etc.). It stipulates a specific relationship between average area A_n of a resting cell and the number n of its edges,

$$A_n = a(n-2) \tag{1}$$

where $a = A_0/4F$, for a tissue containing F cells covering total area A_0. Either one or both of A_0 and F can be time dependent, to account for growth and cellular division; yet the relationship is valid throughout the development of the tissue. It is shown in (Rivier and Lissowski 1982) that the Lewis law is the natural consequence of the equilibrium between entropy and organized form, in other words, the Monte Carlo generated mosaics, for instance the Voronoi polygons of Poisson distributed seeds would obey the Lewis law as well. It also suggests that if it's invalid given the specific mosaic, there must be other constraints, e.g. the mechanical or biological law, that governs the mosaic. In some cases that due to the curvature of the images the area might not be precisely measured, there's also one topological constraint, $\sum(6-n)F_n = 0$, where F_n denotes the face with n edges. Moreover, curvature adjusted area can be tested by (1) as well.

On the experimental side, basilar papilla was chosen based on the following grounds: 1) It is simpler that the organ of Corti in terms of its anatomical structures 2) the regularity of the mature cellular mosaic pattern is well-known and easy to obtain 3) the availability of molecular markers, i.e. anti-cingulin and anti hair cell antigen staining markers. 4) the availability of clones of chick neurogenic genes, one *Delta* homologue.

Apart from the mosaic patterns being observed in the sensory organ, the auditory hair cells attract attention also because the regeneration ability observed in amphibians and fish. In mammals, hair cells appear to produce only during embryonic development(RUBEN 1967), yet ears of fish and amphibians add hair cells continually as the body grows(Popper and Hoxter 1984). For instance, Corwin(Bock and Whelan 2008) reported in sharks, mitotic divisions in epithelial can generate over a million hair cells during the course of post embryonic life.

1.1 Theoretical underpinnings of the phenomenon

One of the main areas of research in developmental biology seeks to understand the key processes and mechanisms that underlie morphogenesis, the formation of structure and form within the embryo. As summarized in (Maini 1996), there were 2 main classes of models. In one, it is hypothesized that a chemical prepattern is set up, either via a source-sink model, or due to diffusion-driven instability(DDI) in a reaction-diffusion system(Turing system). It is then proposed that cells interpret this prepattern by differentiating only where the chemical (or morphogen) concentrations lie above or within specified threshold values. Hence, the observed spatial pattern of cell differentiation is thought to overlie the chemical prepattern. In the second, it is proposed that due to the mechanochemical interactions between cells and their external environment, a spatial pattern in cell density arises. It is then proposed that cells in high density aggregates differentiate.

reaction-diffusion system During recent years the discrete cellular models gain popularity due to the fact that a more detailed characterization of cells is needed. Cells are known to exchange signals as well as matter in a regulated fashion through membrane-based interactions with their immediate neighbors. Discrete cellular models are closely related to the coupled map lattices. The coupled map lattice is a dynamical system with discrete time, discrete space, and continuous state variables, described by a set of coupled finite difference equations, each equation describing the local mapping within the cell and the coupling of the cell to some subset of other cells. In (Plahte 2001) the author has differentiated two types of reaction-diffusion systems, the diffusion drive(DD) and signaling driven(SD) respectively. By acknowledging the fact that diffusion is the most important transport mechanism through gap junctions, one could attribute the phenomenons majorly to the juxtacrine signaling. Therefore, such models are also denoted as cell contact dependent signaling. SD applies to systems where a cell interacts with other cells through the ligand receptor binding processes. One well-studied process of this kind is the Notch signaling pathway.

Delta Notch intercellular signaling In Notch signaling pathway, notch receptor and ligand mediated intercellular signaling has long attracted mathematical modeling. One classic model was raised in(Collier et al. 1996). Focusing on a patch of cells(n$\leq$100), they declared two parameters of most importance: N, the intensity of the inhibition the cell experiences, D, the inhibitory signal that it delivers to its neighbors. Their model is of DD type, and descriptively Delta activity leads to Notch activation of one cell's neighbor, subsequently Delta inactivation of the neighbor, and this leads to Notch inactivation reciprocally, and finally the intracellular mechanism balances itself to adjust Delta activation accordingly. Quantitatively, given one cell, its N increases as the increase of D of its immediate neighbors; its D decreases as the increase of its own N. The fate of the cell univariately depends on N. Therefore, one could imagine the homogenous solution, i.e. isotropic cellular N across the patch, in which inside each cell N balances D perfectly, and such balance is seen across the whole patch. Unfortunately, such balance is delicate and deemed unstable, for had the state of one single cell jiggle a tiny bit off that balance, say N_{this} larger than D_{this}, intracellularly, per the aforementioned quantitative rule(D decreases as the increase of its own N), then D_{this} will decrease even more as the increase of N_{this}. Therefore the patch is unstable even just analyzing this one single cell. What if N_{this} is smaller than D_{this} this time, will the cell return to the stable state again? Seems so, but if we use another aforementioned quantitative rule, N increases as the increase of D of its neighbor, we see that since this time D_{this} has increased, $N_{neighbor}$ must increase as well, then for that neighbor, the situation is again back to the case one. Therefore, the system cannot find a balance again once a cell is perturbed. Hence the conclusion reached by the author, the homogenous steady state is unstable if the feedback is sufficiently strong. But the author found another conclusion, when the homogenous steady state is unstable, the system has a pair of inhomogeneous steady state. It is as if there's a switch that determines whether the cell opts to lower its Delta or its Notch, then higher another, but not all at the same time. Since also the fate decision of the cell depends on N, then we can say that low N leads to adoption of the primary fat and high level leads to adoption of the secondary fate, hence a pair of inhomogeneous steady state.

In (Sprinzak et al. 2010) modelers have incorporated new experimental evidence. The Delta ligand have two activities, it transactivates Notch in the neighboring cells and *cis*-inhibits Notch in its own cell. It's plausible to think about two mutually exclusive modes of cell, the sending mode(high Delta/low Notch) and the receiving mode(high Notch/low Delta). The switch between the two is said to be ultrasensitive, which means a high Hill coefficient that reflects how small change can be amplified into a large change. The ultrasensitive switch is not bistable, meaning it's possible for cell to switch from the sending mode to receiving mode, but not the other way around. This is not the case for the following Corson's model.

(Corson et al. 2017) signaling driven model Following similar experimental observations, each cell receives inhibitory signals from neighboring cells, which if expressed mathematically reads $s_i = \sum_j c_{ij} D^* (u_j)$, where c_{ij} describes the distance-dependent coupling between cell i and j, and $c_{ij} = e^{\frac{-d_{ij}^2}{2l^2}}$, functioning as a Gaussian filter of signals. The evolution of cell state u is govern by

$$\tau \frac{du}{dt} = f(u, s) - u + \eta(t)$$

, comparable to Langevin equation. The high u cell will eventually develop into sensory organ precursors(SOP). The cell-intrinsic feedback, in their main model

$$f(u, s) \equiv f(u - s) = \sigma[2(u - s)]$$

guarantees the bistability of cellular fate. σ is one sigmoidal function. If the feedback is taken out of the sigmoidal function, as $\tau \frac{du_i}{dt} = f(s_i) - u_i$, the model will degrade into being monostable. Dynamically, setting all u to zero

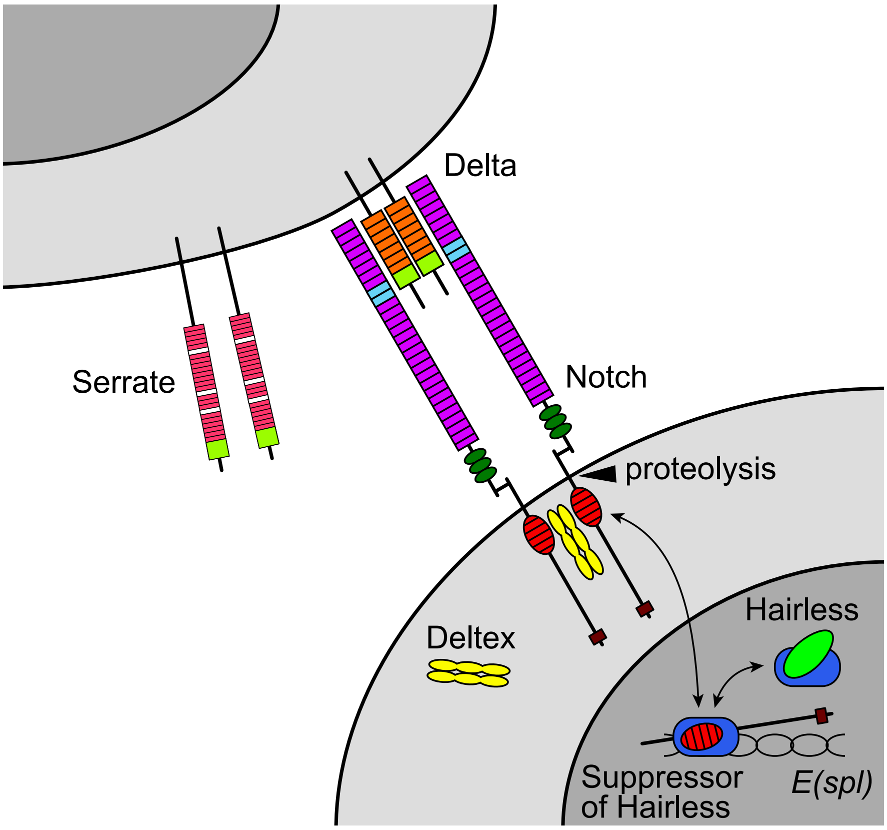


Figure 1: The schematic of Notch juxtacrine signaling between two cells. DSL(Delta/Serrate/LAG-2) family protein is the Notch ligand. Notch receptor is a bipartite protein, composed of a large extracellular domain linked to smaller transmembrane and intracellular domain. Binding of ligand initiates the cascade, including proteolysis, the libration of intracellular domain and the entering of nucleus to engage other DNA-binding proteins and regulate gene expression. Mammals possess four different notch receptors, denoted as NOTCH1, NOTCH2, NOTCH3, NOTCH4.

at the start, because $\sigma'(0) = 1$, which is unstable, all the cells evolve towards a higher u, at the same time increasingly signaling other cells with inhibitory Delta. At about $f(u-s) = u$, all cells plateau, which is a noticeable feature of this bistable model. But thanks to the noise term $\eta(t)$ which is of the strength $\sqrt{2D}\tau$, some cells evolve into SOP, whereas others adopt the epithelial fate. Their model has captured the stripe and dot patterns on a tissue-wide scale.

vertex mechanical model of epithelial morphogenesis The vertex model of epithelial morphogenesis found its origin in (Honda and Eguchi 1980) pondering cell boundary contraction in a monolayer cell sheet. From the energy based perspective, each vertex i in a cellular lattice mesh is imbued with a energy function to minimize,

$$U_i = \sum_{\alpha=1}^{3}(\frac{K_\alpha}{2}(A_\alpha - A_\alpha^{(0)})^2 + \frac{\Gamma_\alpha}{2}P_\alpha^2) + \sum_{j=1}^{3}\Lambda_{ij}l_{ij}$$

in which the K is the parameter related to area contractility, $A^{(0)}$ the area normalization parameter, Γ the parameter related to perimeter elasticity and Λ the parameter related to vertex-vertex junction tension. To calculate energy, sum over the area and perimeters of 3 joining cells and 3 vertex junction length, multiplied by its corresponding parameters. By solving the equation of motion for each vertex $\eta\frac{dx_i}{dt} = -\nabla U_i$, the movement of vertices is tractable. To minimize the energy, it's equal to put it as that each vertex to minimize the additive gradient imposed by each joining cell and neighboring vertex. Then for computational convenience, it's easy to derive the vector direction to minimize area, perimeter of one polygon, which is shown in the

Observable notebook.

Another feature of vertex model is the topological processes, denoted t1 and t2 process respectively. T1 process, the topological rearrangement process of the first kind, is one rule imposing on the mesh that prevents the edge length below a certain threshold, denoted d_{min}. Four faces/cells involved in such process and once t1 process is performed two cells will lose one vertex of its own and another two will gain one vertex. In terms of vertices, apart from two vertices performing such process, there're two additional vertices undergoing topological changes. These two can be found by the following process. If we assume the edge whose length below the threshold will rotate clockwise during t1, namely the two vertices joining the edge will rotate clockwise. If we denote the two vertices v_1 and v_2, and the two cells that going to lose vertex C_1^- and C_2^-, in which v_1 is on the boundary of C_1^- before t1, then v_1' is the next vertex succeeding v_1 in C_1^-, aligning all vertices of C_1^- in the same clockwise fashion. Similarly, v_2' is succeeding v_2 in C_2^-. v_1' and v_2' are the two additional vertices undergoing topological changes. Another parameter involved in t1 process is the separation ratio. Separation ratio can be set to 2, i.e. the new edge formed will be two times the length before t1. But on some occasions where edge length is much smaller that d_{min}, it would be cumbersome to perform multiple t1 for the same edge, therefore the separation ratio can conveniently be set to d_{min} over edge length, and the edge length value is below the threshold. Another handy treatment is to let the new edge formed perpendicular to the previous one, so that it's more symmetric even though we have to choose the clockwise orientation arbitrarily in the first place. It's easy to mathematically derive the coordinates of

v_1 and v_2 after t1, using separation ratio and d_{min}. T1 process preserves the number of cells and number of vertices. Each vertex obeying the equation of motion will move continuously, but t1 process will change the motion of vertex abruptly, during which the motion is not continuous. T1 process will relax the mesh overall, and it's critical to the edge exchange between cells.

T2 process, also involving topological changes, don't preserve neither cell number nor vertex number. When the area of a cell is below certain threshold, how many edges does this cell own is no longer of concern, so it's permissible to treat the small cell as a vertex. It can be said that a cell degrades into vertex. The whole mesh will lose the amount of vertices of the edge number of the cell, and it will also lose one cell, but add one new vertex to replace the cell. The new vertex locates at the centroid of the old cell, which can be found by the shoelace formula. All the neighboring cells will lose one vertex and thus gain one edge, which is guaranteed by Euler's polyhedron formula.

can coupling morphogenesis to a simple fate-specification model generate a zoology of lattice structures?

> In contrast, when mechanisms are coupled mode selection is restricted by the necessity for coherent patterns of broken symmetry between different mechanisms, thus constraining the choice available. (Newman and Comper 1990)

(Goodwin, Kauffman, and Murray 1993) has introduced a thought experiment in which the cell sorting and Turing systems are coupled, and they asked about whether such coupling would yield a more robust patterning system overall. The key insight from it is that by coupling these two, each mechanism breaks the symmetries of the other, resulting much fewer patterns, thus more robust the system is. Their conjecture gave possible explanations to the observed robustness of different development systems. Echoing such thought, (Montévil and Mossio 2015) have introduced some formalism touch in which biological organizations can be considered as a closure of constraints. By their definition, processes refer to the whole set of changes(typically physical processes, chemical reactions, etc.) that occur in biological systems and involve the alteration, consumption, production and/or constitution of relevant entities. This is comparable to the ideas of thread in computer science(Fig 1). Constraints, by their definition, refer to entities which, while acting upon these processes, can be said *from the appropriate viewpoint* to remain unaffected by them. The closure is a set of constraints in which each constraint is both dependent and generative. A closure is said to be strict if the closure is not reducible. The definition again is comparable to the ideas of functional programming, in which a closure is the combination of a function bundled together(enclosed) with references to its surrounding state(the lexical environment).

Taking these underpinnings into account, it's natural to ask by coupling Corson dynamics and the vertex model, whether a stable parameter domain can be obtained, and subsequently a robust dynamic system. The question also permits the possibility that the combinations of parameters could give rise to different lattice configurations, which is promising explaining the variation of hair cell/supporting cell contact number of chick cochlea.

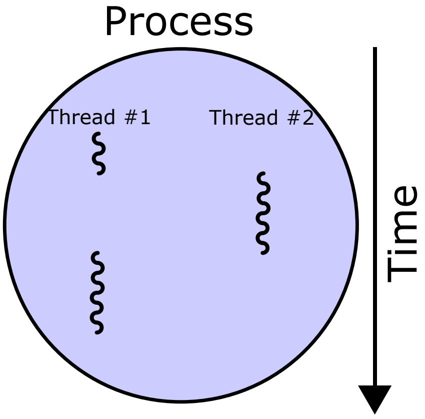


Figure 2: A schematic representation of a process with two threads of execution, running on one processor

1.2 Computer models

One guiding principle of the development of molecular imaging has been that it enables experimentalists to observe the phenomenon in real time, for example the capability of fMRI of showing the blood flow changes in patients' brain, and in fluorescent and confocal microscopy the fluorescent stains revealing the dynamics of nucleic acid, actin fibers, tubulins and other fluorescent proteins. Following this, given the increase of computational capacity and the novel design of computational units, it's highly probable that we can recapitulate some basic biological processes *in silico*. For instance, the NVIDIA Turing architecture combined multiple types of specialized processor core, with enhanced computational precision. It's potentially useful for performing molecular simulations in a more accelerated fashion. The rendering of some delicate fluid dynamics simulations has been shorten to several minutes per frame, for example the Fedkiw lab has demonstrated several physical simulations ranging from the stress and strain of cloth of a dancer, the hair dangling of the shaking head, the smearing of paint on multiple surfaces, to the baking of bread and cookies in an oven. Given such impressive computer representations, one believing in reductionism must wonder the basic building blocks of it, and what can be learned from some models of computer science for the benefit of biological cellular simulation, so in the following paragraphs we shall take a retrospect of some basic standards and models that permit text annotation, scripting, visualization, interaction, and in all confer an efficient and interactive simulation.

SVG

> I want some kid at Stanford to be able to cure cancer in his dorm room. — Steve Jobs, on the purpose of the new NeXT cube

Scalable Vector Graphics(SVG) is an open standard developed by World Wide Web Consortium(W3C) since 1999. Several years before, Macintosh pioneered the development of Adobe Illustrator, which is the most successful commercial software for manipulating SVG to date. Back in 1985, Apple co-founder Steve Jobs intended to sell Macintosh personal computers to universities. At a luncheon held in Silicon Valley, he met the Nobel Laureate in Chemistry Paul Berg. Berg was frustrated by the expense of teaching students about recombinant DNA from textbooks instead of in wet laboratories, therefore he suggested Jobs to create a ”3M computer” workstation for higher education, featuring at least one megabyte of random-access memory(RAM), a megapixel display and megaFLOPs performance. This precluded the advent of Jobs’ new company, NeXT, Inc. To some extent, this 5-year-aheadof-time personal computers, utilized by Tim Berners-Lee at CERN, gave rise to the world's first Web server and ran the world's first Web browser in 1990. Some primitive standards of Internet and the underlying framework of browsers thus came into birth, for example the so-called three pillars of world wide web(HTML, JavaScript, CSS), and the lately-come SVG in 1999.

SVG is an efficient representation of graphics in computers. The most notable feature of which is scalability. This means that the images can be rendered as any size without loss of quality, compared to classic bitmapped image formats such as JPEG or PNG. Therefore, scaling the bitmap reveals the set of pixels whereas scaling the vector image preserves the shape. SVG images and their related behaviors are defined in XML text files, which means they can be searched, indexed, scripted and compressed. Additionally this means they can be created and edited with any text editor or with drawing software. For example the following snippet in HTML markup language shows how to draw a polygon on a modern browser. One character of HTML is the tagged format with angle brackets, and the elements are surrounded by matching opening and closing tags. Each element starts with the type of the tag, for example `svg`, `text`, `polygon`, followed by several attributes that are characteristic to its own kind, say for polygon `points` attribute gives the polygon anchoring position on the display, and `fill` attribute specifies the color filling of its closure and `stroke` attributes specify the color of its boundary.

```
<svg viewBox="0 0 200 100" xmlns="http://www.w3.org/2000/svg">
 <polygon points="100,100 150,25 150,75 200,0"
          fill="none" stroke="black" />
```

```
</svg>
```

Listing 1: W3C standard for drawing polygon in SVG

The `polygon` element lying in the scope of `svg` element implies a hierarchical structure of several objects being manipulated. Once a single polygon is drawn, adding another polygon won't disrupt the previous polygons drawn, and in DOM(Document Object Model) representation this means simply adding a new branch to the tree, with its own id and corresponding attributes. A container of objects is the node, lying in the ends of the tree. The reason for having a container for objects is to manipulate those objects as a whole, for example the sets of polygons. The event handlers can also be attached to nodes, listening for the possible events like clicking, panning, zooming, and thus such model permits interactivity. All the shapes are contained by the canvas element, which is part of the introduction of HTML5 and allows for dynamic, scriptable rendering of 2D and 3D shapes and bitmap images.

For optimized speed of rendering shapes, the shape-rendering attribute can be set to `optimizeSpeed`, which is to emphasize rendering speed over geometric precision and crisp edge. This option will sometimes cause the user agent to turn off shape anti-aliasing.

asynchronous function Asynchronous function is one feature of JavaScript ES6, which break the usual linear execution of an event loop and let the loop deals with multiple functions at the same time. For example, if you go down to a fast food joint and order a cheeseburger they will immediately take your order and make you wait around until the cheeseburger is ready. In the meantime they can take other orders and start cooking cheeseburgers for other people. Imagine if you had to wait at the register for your cheeseburger, blocking all other people in line from ordering while they cooked your burger! This is called blocking the event loop because the whole event loop (enter food joint, order, cooking cheeseburgers) happens one at a time. By using async functions, which operate in a separate order than the rest of the code, the functions become non-blocking, which means it can cook many cheeseburgers at once. Under the hood, by declaring one function async, the function will return an implicit promises as its result. A promise is a wildcard for the code to look more linear, a proxy for a value not necessarily known when the promise is created. It can also be used to delay a loop's running of codes, as demonstrated in the main simulation loop 2 later on.

the Observable notebook Observable notebook is one instance following the ideas of REPL(Read-eval-print loop). For example in MATLAB console when typing in an arithmetic expression after the quick evaluation the output would be printed following the input. Similarly, in Jupyter notebook a graphics element will output below the input command. In Observable notebook the ideas follow but due to the event listeners in DOM different computing cells form hierarchical dependencies. Once a cell is edited then an event will fire and propagate, this ends up all dependent cells reevaluating itself. Moreover, since the notebook lives in the web it grants the greatest accessibility amongst all computational notebooks. The canvas element is hardware accelerated in modern browsers, therefore some graphics-heavy tasks can also be performant.(One could type `about://gpu` in the address bar to get the graphics feature status implemented in your browser) Lastly Observable notebook incorporates git version control system innately, therefore every notebook can be forked(not the utensil) and merged later on, facilitating cooperating among communities.

modeling environments previously employed One computational framework/library related to the vertex mechanical model is Chaste(Cancer, Heart and Soft Tissue Environment). Chaste is inclusive and extensible, and focuses on three: 1) continuum modeling of cardiac electrophysiology(Cardiac Chaste), 2) individual-based modeling of cell populations, with specific application to tissue homeostasis and carcinogenesis(Cell-based Chaste), 3) reduced dimensional modeling of ventilation in lungs(Lung Chaste). Cell-based Chaste has incorporated the classic Delta Notch signaling in its library. On the software aspects, Chaste is based on multiple dependencies, including MPI, PETSc, HDF5, VTK and so on, therefore the easiest way to install Chaste is to use Ubuntu Linux operating system or a docker image of the similar system. For the extensibility, a prior knowledge of various C++ features, including `std::vector`, `c_vector`, and object-oriented inheritance including abstract classes and virtual methods is necessary. The computational process is initialized by CMake commands, and the result can be visualized using ParaView. For a vertex model simulation of ∼100 cells, the calculation takes approx. 2 mins. However, on the realistic side, bridging the parameters across different modules of cell-based Chaste is hard, for it has been divided into three modules a priori: 1) module of cellular behavior(the cell cycle model, programmed cell death), 2) module of movement and mechanical interaction 3) module of the transport of key nutrients, signaling molecules.

Another way of constructing and running simulations of the broad scientific community is MATLAB. However, the major downside of running vertex model simulation in MATLAB is that it's slow to plot multiple graphic elements in one frame / one loop run simultaneously. Empirically, for ∼100 cells the performance is around 1 FPS(frame per second), taking together the computation and plotting. For any improvements on it, it would require NVIDIA CUDA-enabled GPUs. In other words it precludes MacOS and additionally high-end GPU has to be at hand.

2 Methods

All the experiments are performed *in silico*, and this section will include mostly coding/programming languages and data structure relevant details of the project.

random seeds and the maximum arbitrariness of meshes The first step of the simulation is generate some random seeds in the 2d plane. There're two ways of doing so. Hardcore packing algorithm is to treat every seeds as a 2d circle with radius r, and one seed has partitioned some space in the 2d plane it will prevent another circle from intersecting with itself. It is as if each seed is a rigid 2d ball and the random number generator is employed find one possible packing state of all balls, hence hardcore. Another way that it's more efficient in searching the packing state than hardcore algorithm is the Poisson disk sampling(Dunbar and Humphreys 2006). The resulting points are tightly-packed, but no closer to each other than a specified minimum distance.

After generating the seeds `d3−delaunay` is used to find Delaunay triangulation. Each circumcenter of adjacent triangles is then connected and the Voronoi diagram is constructed. In MATLAB there's one built-in function `voronoin`, which is based on Qhull (Barber et al. 1996), the quickhull algorithm for finding the convex hull of points. Qhull is a comparatively old implementation of the algorithm in C/C++, dated back to 1996, and yet with the thriving ecosystem of web developers, JavaScript(JS) reached maturity with the publication of ECMAScript 6(ES 6) in 2015, and thus come a new implementation of triangulation algorithm in JavaScript, named Delaunator. A JS library called D3 has adopted Delaunator and it's fast and run without dependency in Observable notebook. But there's a lack of computational speed benchmark between quickhull algorithm and sweep algorithm employed by Delaunator.

class of objects For graphs, there are two basic ways to represent them in arrays, one is the adjacency list, which describes the set of neighbors of a vertex in the graph. For example, in a graph where vertex a connects to vertex b to vertex c and back to a and thus form a connected graph, we write $\{b, c\}, \{a, c\}, \{a, b\}$. Another way is adjacency matrix, which is a table of all edge-vertex incidences. Adjacency matrix can be applied globally, meaning that the whole vertex mesh registers one matrix, but it can also be applied locally, that each vertex only possesses the information it needs. For vertex model simulation, each vertex only needs to know its three neighbors that it shares an edge with, and three neighboring cells that join to form itself. That is its own topological information. It is supposedly a very efficient data structure to query and update.

An object, in terms of object-oriented programming(OOP), will possess the properties of its own, for example the neighbor vertex of in the format of an array. An object will also possess its own methods, which is a procedure to derive its own properties. In JS, an object will have a notion of itself, denoted as `this`, and it's used inside one specific class to refer to the current class, and is only accessible within the specific class. Object also has a special method denoted `constructor()`, which is used only once for creating and initializing this object within specific class. One JS feature is that it's not necessary to declare the property beforehand in order to assign some value to this property, but we can declare any necessary property on-the-fly and query them later on.

A vertex object possesses the following properties: index(idx), joining cell(jCell), neighboring vertex(nVtx), location(loc), whereas a cell possesses the properties: index, its vertices(which refers back to the corresponding vertex object, shorted as vtx), area, perimeter(these two need updating every time the position of vertices is updated and t1/t2 processes performed), centroid, parameter K, A_0, Γ, cell state u as in the Corson model. Then there are vertex set and cell set, containing all the vertices and cells. Till now we have four classes, vertex, vertex set, cell, cell set, and each class has some non-trivial methods. For vertex class, the method is `setForce()`. It queries its own location and nVtx to find their locations, in order to calculate the edge junction term of the force. It also queries jCell to calculate the perimeter and area contribution of the force. Lastly sum them up to set a force property to any given vertex. For vertex set class, the non-trivial method is `t1check()`. At high level it performs the job as expanded in the introduction section under the paragraph vertex model, and throw any exceptions should unexpected error occurred. Lastly for class cell set, there are two major methods. One is `t2check()` which is similar to t1 process as analyzed before, and another one is `corsonSignal()`. Corson signal is defined in cell set but not cell class because Corson model is of SD type, that a mesh needs treating as a whole for the patterning to happen. Additionally there's no property maintained for cell object to find its neighbor, therefore should any necessitates, a global search has to be made for finding all the neighboring cells, or the new property should be declared and maintained during the class's own construction.

A simulation class is used to construct aforementioned four classes, using the output of voronoi diagram of D3 library. Every time a new simulation initializes this class constructs itself, returning an object with property `v` containing the vertex set and property `c` containing the cell set.

The main simulation loop is organized as follows. Vertex set will first update all the positions of vertices due to the force vector has been calculated for the initial state, then cell set will update its area and perimeter. Then do the t1 and t2 check(sort of like t1 and t2 daemon respectively). Find the force for all vertices. Let Corson signal ensue. Use the corson state to change the cell's mechanical parameters by any potential functions. Lastly upload the state to canvas for rendering. An async delay is imposed on the loop to control the speed of simulation(to slow down the simulation, but it can't accelerate the simulation), and yield is used to pause and resume updating the canvas, in accordance with the use of promises.

```
{
 while (simulate) {
   await Promises.delay(sst(ss))
   state.v.updatePosition()
   state.c.update()
   state.v.t1check()
   state.c.t2check()
   state.v.setForce()
   state.c.corsonSignal()
   state.c.forEach(e=>e.setK(function(e.u)))
   yield canvas.update(state)
 }
}
```

Listing 2: The main simulation loop

periodic boundary condition The last part of methods deals with periodic boundary condition(PBC). PBC, if interpreted intuitively, in 1D is equivalent to an infinite line of cells in which alternate cells are identical. In 2D, the infinite line is extended into infinite 2D plane, and each patch of cells(the minimum repetitive unit) exhibits translational symmetry. There are many approaches for imposing PBC, for example mapping the locations of all simulated objects to a toroidal curved surface where the periodic boundary condition naturally exists; cell image can be created if cells are close enough to boundary so that once cells move out of the boundary they can be forced to reappear at other end of the boundary. The downside of the first approach is that cells in the outer rim of a torus get enlarged whereas those in the inner rim get contracted, thus the graphics representation will be curvature biased. For the second approach, more cell instances have to be initiated and bookkept.

In the first step of generating random seeds, considering PBC, the seeds confined to a rectangular box have to be copied 8 extra times, surrounding the original shape. The exact space to be copied is not a compact space since out of four edges of the rectangle the boundary points of two are not included. And out of four vertices of the rectangular box only one vertex should be included. For any given seed in the box, there are 8 copies in the surrounding area. This can be optimized by diving the box first then copying box to make the rim. One way is to divide the box into 9 pieces, labeled 1 to 9 from the upper left to the lower right respectively(go by the row first, finishing one row and then to the next row). The 9 smaller boxes are of the same area. Then from box1 to box9, in order to make the rim, we only need to copy the seeds 3, 1, 3, 1, 0, 1, 3, 1, 3 times respectively. In this way, a vertex will have at most 3 cousins living in the rim, and at least none. It's unnecessary to have more cells apart from those in the original box, and when updating vertex location it means updating all the vertices including the copies by the same displacement. In this way, it's straightforward to calculate the force for vertices in the box, since all the vertices in the box will have surrounding vertices. The joining cell of those vertices near the boundary will have the cells mapped back to the cells in the box. During t1 and t2 process, one function that's extremely useful is to find closest vertex to certain cell out of vertex.ring, and vertex.ring here denotes the union of the vertex itself and those copied vertices(vertex.ring has the length at most 4 and at least 1). Empirically this is the most efficient way to impose PBC that I have known of.

3 Results and Discussions

> Pattern is recognized by naked-eye observation, but not image algorithm, therefore to generate the near-plausible graphics using our computers seems imperative.

qualitative interpretation of the three parameters involved The steady state can be reached for the set vertices, if K, Γ and λ are not zeros at the same time, and the steady state exhibits a fast switch between two states, as observed in the simulation. Next follows the description of the specific influence of the three parameters on the mesh.

K The direct interpretation of this parameter is the weight over minimizing cellular area. The mesh under the sole influence of this parameter exhibits general stress and relaxation, and the general shapes of cells are preserved. Sometimes the convexity of individual cells cannot be guaranteed if there're no any other parameters involved, and if such case occurs, the mesh cannot reach the steady state.

Γ The direct interpretation of this parameter is the weight over minimizing cellular perimeter. The steady state under the sole influence of this parameter is oscillatory between two stable states for set of vertices, while topologically every cells tend to form equilateral polygons, since $-\nabla P$ bisects the internal angles. This subsequently will polarize the areas of cells.

λ The direct interpretation of this parameter is the weight over minimizing cellular junction length. The steady state under the sole influence of this parameter will vanish into a single vertex, since it will minimize the length of any edges available.

two classes of cells The general goal is to let Corson dynamics, i.e. the selection of hair cells influences the topology of vertex mesh so that a regular tiling pattern could emerge. The first experiment I performed is to use a linear function $f(u) = 1 - u$ to let cell state u influence K. A high u cell will have a low K, and vice versa. Then at steady state the area of hair cell(high u) can be controlled by the value of A_0. A lower than average cell area A_0 will dilate the hair cell and a higher A_0 will shrink the hair cell. By using this technique and by adjusting the signal potency parameter, i.e. a low signal potency will let more hair cells emerge, a high potency will let fewer hair cell emerge, one pattern that's comparable to the experimental observation made by (Goodyear and Richardson 1997) can be produced (Fig.3). Also note that slight increase of Γ will make cells look more regular.

Since the patterning is not obvious compared to regular tiling patterns in Euclidean plane, some statistical measures might be of help. Preliminarily,

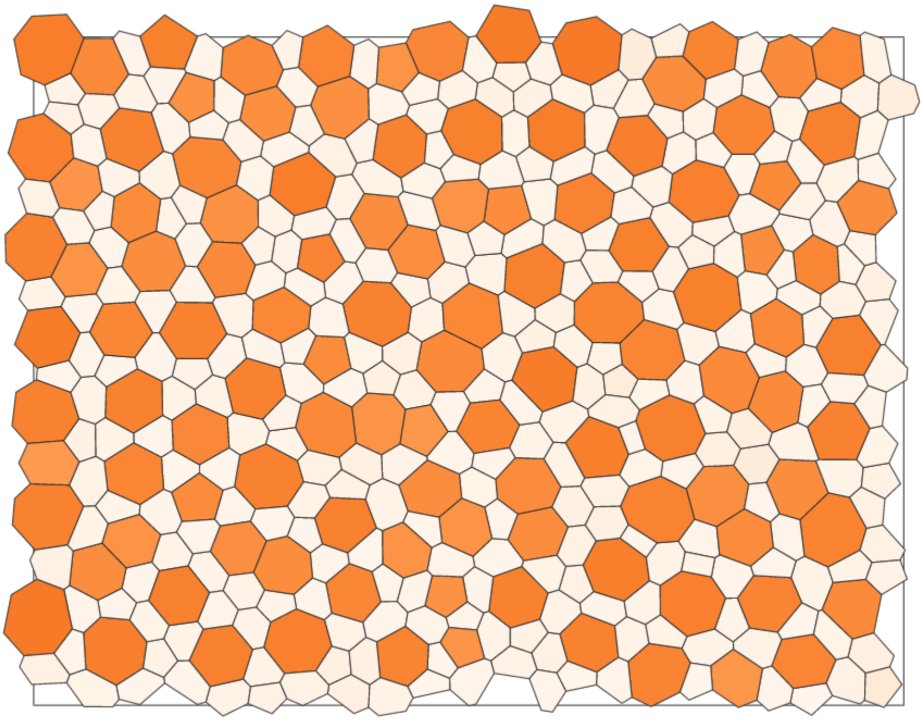


Figure 3: Computational simulation steady state of hair cells in orange interspersed by supporting cells, comparable to the maturity of hair cell emergency at E12 in(Goodyear and Richardson 1997), here $S/H = 2.58$.

there are six:

1. how many hair cells contact each other (HH)
2. how many hair cells are interspersed by one supporting cell, in other words, two hair cells are neighbored by one single supporting cell (HSH)
3. global supporting to hair cell ratio (S/H)
4. hair cell contact number, i.e. how many supporting cells does a specific hair cell contact (HS)

5. supporting cell contact number, i.e. how many hair cells does one specific supporting cell contact (SH)

6. hair cell edge distribution ($e(H)$).

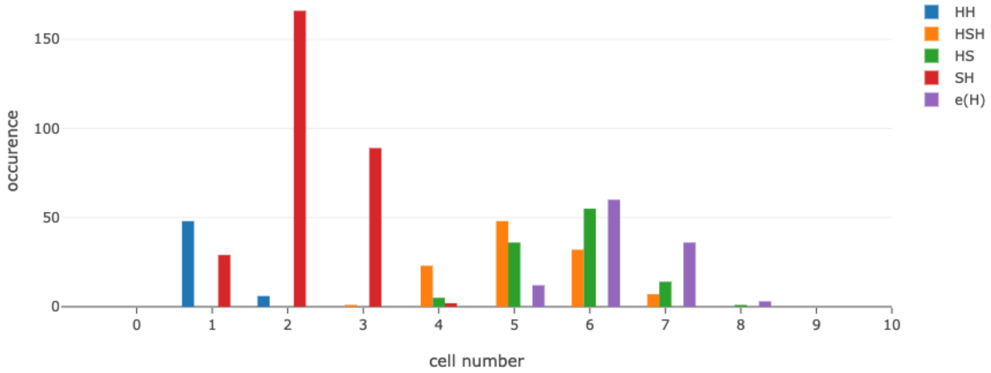


Figure 4: Five topological measures corresponding to Fig.3, discussion seen in the texts.

Fig.4 has shown the five measures of the same mesh configuration. According to the experiment, for the red bar the largest occurrence lies in number 3 for central-distal(CD) and superior-proximal(SP) area, whereas for inferior-proximal(IP) number 2 has the largest occurrence. Therefore, judging from red bars, namely how many hair cells a supporting cell contact, Fig.4 is most likely reflecting the experimental observation of IP. Also for IP, number 3 is second largest occurrence, which matches here. Yet if we look at green bars, which is rightly the reciprocal scenario(HS compared to SH), the most largest occurrence lies at 6, and it corresponds to statistics of CD and SP area in E12, the near-maturity development day of chick cochlea.

For IP, the distribution leans more towards a higher number, that number 9 or 10 are seen as the most frequent HS value in experiment. The blue bars reflect the hair-hair cell contact at steady state. In E12, only distal areas have reported the existent contacts and across the whole tissue the contact is more of a transient state during development, most widely seen at around E8 and E9. The orange bars are a measure of how many most nearest hair cells surround one given hair support, which hasn't been reported by the author. A major weight of this measure on number 5, 6 is acceptable by the raw check of staining images of cells. The slight difference in placement of x axis between green bars and purple bars are due to the hair-hair contact. Lastly, $S/H = 2.58$ is a stat lying between SP(2.14) and IP(3.90) area. Taken together, the simulated configuration is most likely a patch of cells between SP and IP, and yet hair-hair contact should be precluded in the steady state.

Given the analysis, there are two things that can be learned. One is it might suggest that the switch from Corson dynamics to the classic Delta Notch model is necessary for preventing the cases of hair-hair contacts. The second is if the simulated result does reflect some patches between SP and IP, then a more detailed check of the imaging data of the desired measure combinations is needed.

Given the way the computational notebook is organized, we can also tune the signal potency to preclude any hair-hair contact and follow the same analysis pipeline again.

As seen in Fig.5, $S/H = 4.09$ suggests the IP configuration, yet the green and purple bars are still centered around number 6 and 7, suggesting more of a CD and SP configuration in E12. The only way to make it closer to the corresponding IP configuration is to let hair cell dilate more, yet prevent any HH cases from occurring. Also SH weighs between 1 and 2, which is ectopic and not seen in neither CD nor SP nor IP(weighs between 2 and 3 or between 3 and 4). This thus suggests a more stark difference of K between two classes of cells.

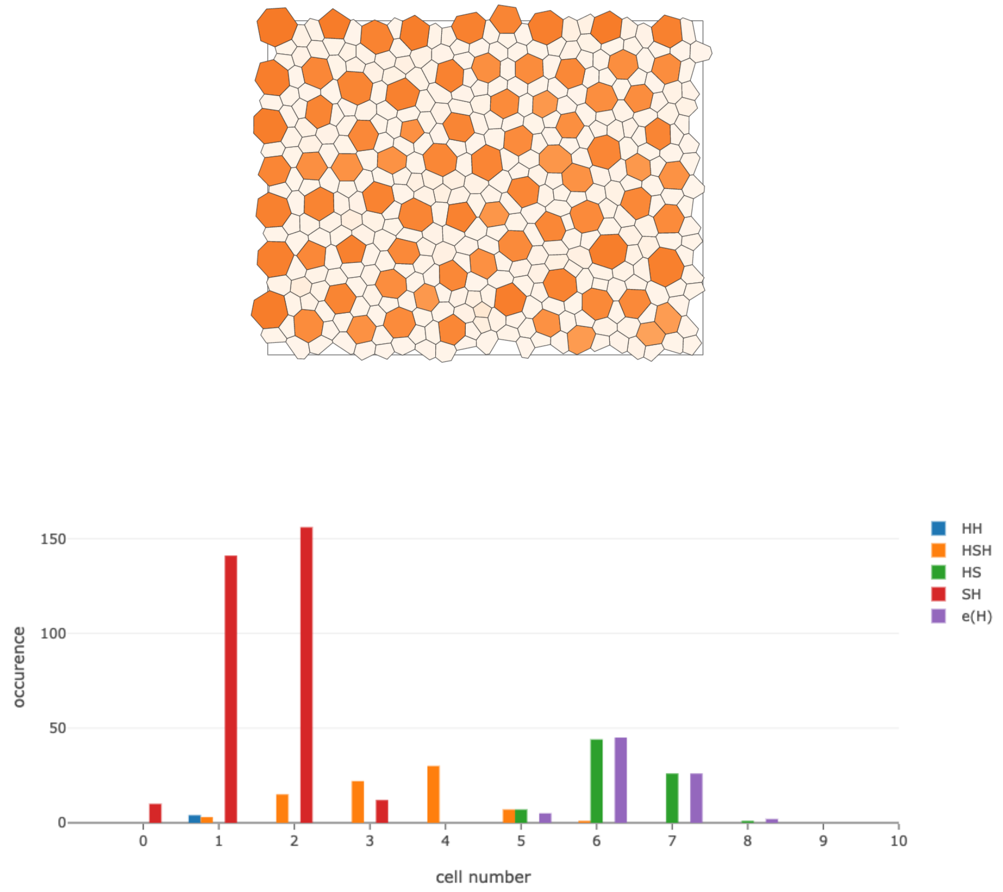
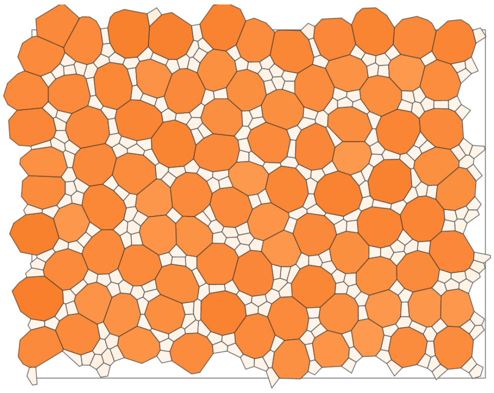


Figure 5: One configuration at the vanishing point of HH, $S/H = 4.09$

nonlinear mapping from u to K Here a nonlinear mapping is used to get Fig.6. Ignoring the presence of HH at number 2 and 3, we have seen that the purple bars weigh over 9, 10, 11, which is a good fit of IP configuration.

$S/H = 3.49$ corresponds with IP at E9(3.56) very well, though a little off E12(3.90). Red bars center around 2,3, which also matches IP configuration. In all, this a good fit of IP ignoring hair-hair contact.



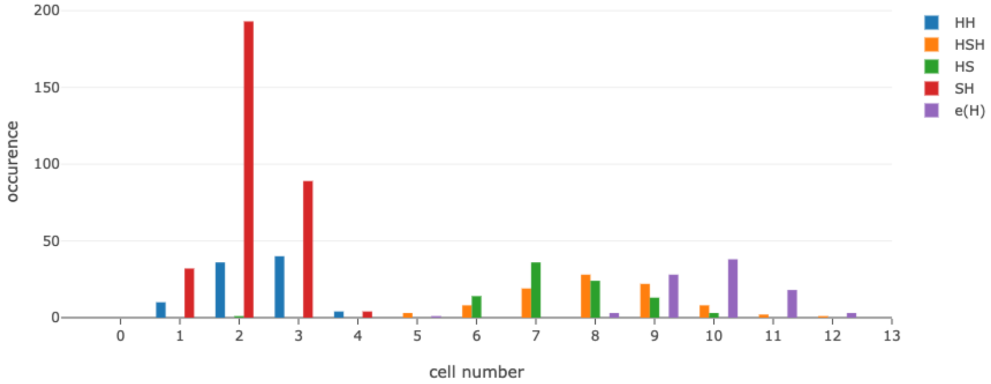
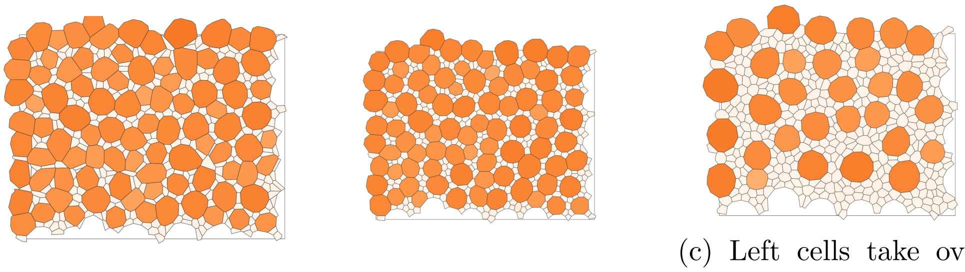


Figure 6: Patterning set $K = (1 - tanh(2 * u))$, $S/H = 3.49$

There're more patterns that can be generated based on this nonlinear mapping. Majorly one can tune the signal potency at steady state to get a variety of patterns, not limited to the patterns shown below as of finishing writing this thesis.



(a) Sheared configuration

(b) Small cell grows adjacent to large

(c) Left cells take over the space of degenerated cells

Figure 7: Patterns of three

4 Prospect

morphological measure of cells A purely edge number based topological measure is just one facet of the observed phenomenon. IP CD and SP has exhibits characteristic morphology, i.e. the characteristic shapes of various cells as well. Therefore, apart from focusing on contacted edge, and thus derived statistical measures, in order to get a wider picture of the imaging, a robust morphological measure should be brought up. GPU accelerated neural networks can be trained to classify cells based on forms, which is the kind of tasks similar to classifying Chinese characters or handwritten digits.

global shear motion might facilitate the regular patterning (Cohen et al. 2019) has asked about if the hexagonal pattern is obtainable in the outer hair cell region. By simulation, they found that an external forces are needed to compress other cells into a packed state. The laminar flow of supporting cells above the group of hair cells imposes a global shear motion, and hair cells exhibit quadrilateral regularity as a result. Therefore, it's plausible that a global force enacting on every vertex of the mesh will facilitate a more regular patterning compared to the local force depended upon the joining cells and neighbor vertices.

References

Major Sources

Collier, J. R. et al. (Dec. 1996). “Pattern Formation by Lateral Inhibition with Feedback: A Mathematical Model of Delta-Notch Intercellular Signalling”. eng. In: *J. Theor. Biol.* 183.4, pp. 429–446. ISSN: 0022-5193. DOI: 10.1006/jtbi.1996.0233.

Corson, Francis et al. (May 2017). “Self-Organized Notch Dynamics Generate Stereotyped Sensory Organ Patterns in Drosophila”. en. In: *Science* 356.6337. ISSN: 0036-8075, 1095-9203. DOI: 10.1126/science.aai7407.

Goodyear, R. and G. Richardson (Aug. 1997). “Pattern Formation in the Basilar Papilla: Evidence for Cell Rearrangement”. eng. In: *J. Neurosci.* 17.16, pp. 6289–6301. ISSN: 0270-6474.

Rivier, N. and A. Lissowski (Mar. 1982). “On the Correlation between Sizes and Shapes of Cells in Epithelial Mosaics”. en. In: *J. Phys. A: Math. Gen.* 15.3, pp. L143–L148. ISSN: 0305-4470. DOI: 10.1088/0305-4470/15/3/012.

Sprinzak, David et al. (May 2010). “Cis Interactions between Notch and Delta Generate Mutually Exclusive Signaling States”. In: *Nature* 465.7294, pp. 86–90. ISSN: 0028-0836. DOI: 10.1038/nature08959.

Minor Sources

Artavanis-Tsakonas, Spyros (Apr. 1988). "The Molecular Biology of the Notch Locus and the Fine Tuning of Differentiation in Drosophila". English. In: *Trends in Genetics* 4.4, pp. 95–100. ISSN: 0168-9525. DOI: 10.1016/0168-9525(88)90096-0.

Bock, Gregory R. and Julie Whelan (Apr. 2008). *Regeneration of Vertebrate Sensory Receptor Cells*. en. John Wiley & Sons. ISBN: 978-0-470-51413-9.

Cohen, Roie et al. (July 2019). "Shear Forces Drive Precise Patterning of Hair Cells in the Mammalian Inner Ear". en. In: *bioRxiv*, p. 707422. DOI: 10.1101/707422.

Dunbar, Daniel and Greg Humphreys (July 2006). "A Spatial Data Structure for Fast Poisson-Disk Sample Generation". In: *ACM Trans. Graph.* 25.3, pp. 503–508. ISSN: 0730-0301. DOI: 10.1145/1141911.1141915.

Goodwin, Brian C., Stuart Kauffman, and J. D. Murray (July 1993). "Is Morphogenesis an Intrinsically Robust Process?" en. In: *Journal of Theoretical Biology* 163.1, pp. 135–144. ISSN: 0022-5193. DOI: 10.1006/jtbi.1993.1112.

Honda, Hisao and Goro Eguchi (June 1980). "How Much Does the Cell Boundary Contract in a Monolayered Cell Sheet?" en. In: *Journal of Theoretical Biology* 84.3, pp. 575–588. ISSN: 0022-5193. DOI: 10.1016/S0022-5193(80)80021-X.

Maini, P. K. (Dec. 1996). "Spatial and Spatiotemporal Pattern Formation in Generalised Turing Systems". en. In: *Computers & Mathematics with*

Applications. BIOMATH-95 32.11, pp. 71–77. ISSN: 0898-1221. DOI: 10.1016/S0898-1221(96)00198-8.

Montévil, Maël and Matteo Mossio (May 2015). "Biological Organisation as Closure of Constraints". en. In: *Journal of Theoretical Biology* 372, pp. 179–191. ISSN: 0022-5193. DOI: 10.1016/j.jtbi.2015.02.029.

Newman, S. A. and W. D. Comper (Sept. 1990). "'Generic' Physical Mechanisms of Morphogenesis and Pattern Formation". eng. In: *Development* 110.1, pp. 1–18. ISSN: 0950-1991.

Plahte, Erik (Nov. 2001). "Pattern Formation in Discrete Cell Lattices". en. In: *J Math Biol* 43.5, pp. 411–445. ISSN: 1432-1416. DOI: 10.1007/s002850100105.

Popper, Arthur N. and Becky Hoxter (Aug. 1984). "Growth of a Fish Ear: 1. Quantitative Analysis of Hair Cell and Ganglion Cell Proliferation". en. In: *Hearing Research* 15.2, pp. 133–142. ISSN: 0378-5955. DOI: 10.1016/0378-5955(84)90044-3.

RUBEN, R. J. (1967). "Development of the Inner Ear of the Mouse : A Radioautographic Study of Terminal Mitoses". In: *Acta Otolaryngol. (Stockh.)* 220, pp. 1–44.

Skeath, J. B. and S. B. Carroll (Apr. 1992). "Regulation of Proneural Gene Expression and Cell Fate during Neuroblast Segregation in the Drosophila Embryo". en. In: *Development* 114.4, pp. 939–946. ISSN: 0950-1991, 1477-9129.