Using cellzilla for plant growth simulations at the cellular level

Bruce E. Shapiro1,2*, Elliot M. Meyerowitz3 and Eric Mjolsness4

1 Department of Mathematics, California State University, Northridge, CA, USA
2 Biological Network Modeling Center, Caltech, Pasadena, CA, USA
3 Division of Biology, Howard Hughes Medical Institute, Caltech, Pasadena, CA, USA
4 Department of Computer Science, University of California, Irvine, CA, USA

INTRODUCTION

In recent years, there has been much interest in accurate multi-scale models of morphogenesis. Due to the various levels of complexity and the wide variety of tissue, there has necessarily been a trade-off between generality and specificity; an excellent review is given by Koumoutsakos et al. (2011). Of the most interest to plant biologists, perhaps, are platforms that describe the high-level structure of complete organisms, systems, or organs such as a meristem or sepal. Very few general purpose tools exist at all, and even fewer are specific to plants. L-System based tools, the best known being L-Studio (Karwowski and Prusinkiewicz, 2004), were among the first; they are ideally suited to branching structures because they are based on formal language theory (a grammar based on axioms, a short alphabet, strings derived from that alphabet based on specific production rules that are tuned to branching). L-system rules have been encoded in higher level languages such as C [specifically, cpfg, (Prusinkiewicz and Lindenmayer, 1990)] to allow the encoding of models and the description of geometrical and topological relationships. The Virtual Plant’s OpenAlea Platform (Pradal et al., 2008) provides a general-purpose collection of simulation modules that use Python as the primary scripting language and allows L-systems to be incorporated (Boudon et al., 2012).

Outside of plant biology, much of the interest has focused on grid-based models such as the Cellular Potts Model (CPM) (Graner and Glazier, 1992), which discretize to the desired sub-cellular level of detail as collections of entities driven by particle-particle interactions; and finite element models (FEM) which instead discretize to a sufficiently fine web-work of straight lines under the influence of mechanical forces. Because of the molecular level of detail of these methods they are often primarily stochastic in nature (Gillespie, 1976). There has been some effort to provide general purpose platforms; examples include MCell (Stiles and Bartol, 2001), designed originally to simulate the synaptic junction; Smoldyn (Andrews, 2012), primarily emphasizing nano-scale specificity; ChemCell (Plimpton and Slepyo, 2005), which models protein networks within cells; and CompuCell3D (Cickovski et al., 2007) which focuses on cellular function.

Of particular interest is the use of rule-based models, e.g., BioNetGen (Faeder et al., 2005, 2009) and NFSIM (Sneddon et al., 2011). In a rule-based model, molecules are considered structured objects and their interactions are described by rules for transforming these objects. Rule based models are interesting because every process that occurs within a biosystem, such as chemical and physical interactions and growth, development, and cell division and death, can be described by rules. Similarly, Dynamical Grammars (Mjolsness and Yosiphon, 2006; Yosiphon, 2009; Mjolsness, 2013) define models in terms of operator algebras of stochastic processes, with simulation algorithms.
derived from the composition and expansion of time-evolution operators.

At an intermediate level of complexity there are tools that describe tissue at the multicellular level, treating each cell as a well-mixed compartment. Cells can be described either as simple point (or spherical) objects connected by breakable springs (Jönsson et al., 2004, 2005a,b; Mjolsness, 2006), or at some level of geometric complexity with springs between polygonal vertices (Rudge and Haseloff, 2005; Sahlin and Johnsson, 2010), as is currently done by Cellzilla. The VirtualLeaf (Merks et al., 2011) provides an interesting hybrid that uses the CPM in combination with mechanical springs with a Markovian relaxation algorithm to describe tissue growth. Cellzilla has the distinction among these software implementations of extending the collection of Cellerator Arrows to include multicellular interactions (Shapiro et al., 2003).

**MATERIALS AND METHODS**

**CELLERATOR REACTIONS**

Interactions are described in terms of Cellerator arrows (Shapiro et al., 2003); the basic canonical form is \([X \rightarrow Y, k]\). When several species are interacting the arrow expression is written as

\[
e_1 X_1 + e_2 X_2 + \cdots \rightarrow f_1 Y_1 + f_2 Y_2 + \cdots
\]

where \(e_i\) and \(f_j\) are stoichiometries. Each reactant is converted to a single term in a differential equation according to mass action kinetics:

\[
\frac{dU_i}{dt} = (f_j - e_i)kX^{e_1}_1 X^{e_2}_2 \cdots
\]

where \(e_i\) and \(f_j\) are the stoichiometries of species \(U_i\) in the reaction on the left and right hand side of the reaction, respectively, and \(k\) is the rate constant of the reaction. More complex expressions can be built from this canonical form to represent exact mass action descriptions of multi-species complex enzymatic reactions, as summarized in Table 1, and numerous regulatory approximations such as Michaelis-Menten, Hill functions, and MWC equations, as summarized in Table 2. Furthermore, a large number of exact enzymatic expansions have been implemented using the basic canonical form via the KMech toolbox, including BiBi, BiTer, BiUni, MulS, OrderedBiBi, OrderedBiUni, PingPong, PingPongTerTerF, PingPongTerTerR, RandomBiBi, TerBi, TerTer, UniBi, and UniUni reactions (Yang et al., 2005).

Cellzilla is implemented in Mathematica and is invoked using the standard notebook interface. The details of numerical integration are normally hidden from the user, e.g., models (such as (30)–(32)) are submitted to the kernel by the Cellzilla Grow command and Cellerator run along with a list of simulation control parameters and initial conditions. These front-end commands directly invoke Mathematica’s NDsolve. Normally, the default solver in NDsolve is chosen, unless the user selects otherwise; any control option for NDsolve may be passed along by Grow command. For ordinary differential equations, NDsolve switches between a non-stiff Adams method and a stiff Gear Backward Differentiation formula. However, users may optionally change the parameters or choose another solver.

**TISSUE DESCRIPTION**

Tissues are described by a polygonal lattice, with each lattice cell representing one biological cell. The Tissue data structure consists of: (1) a vertex list \(V\), where each vertex \(V_i\) is an \((x, y)\) pair; (2) an edge list \(E\), where each edge \(E_k = (i_k, j_k)\) is a pair of integers giving indices of vertices at the endpoints of the edge; and (3) a list of cells \(C\), where each cell \(C_k = \{k_1, k_2, k_3, \ldots\}\) is an ordered list of edge indices. Externally the tissues may be saved (either read or written) as CSV files, either as lists of vertices, edges, and cells, or in a flattened versions with the edges omitted and the cells represented as ordered sequences of vertices. Internally the edges are always reconstructed since it is more efficient computationally to always have the edge information available, but it is user taste which I/O format is used. It was decided to use the CSV format for these files because of the wide availability of parsing tools and the ease of human readability that should be necessary.

Species in different cells are referenced by an index; e.g., the reaction \(X[17] \rightarrow Y[17]\) takes place in cell 17. When a reaction network is expanded in every cell in the system it is not necessary to repeat this manually, as this is done automatically. Constituents in different cells can interact in the following ways: (a) diffusion; (b) action at a distance; (c) transport across the cell wall. Each of these may be specified in the model by one of the additional Cellzilla Arrow forms listed in Table 3. Such interactions are allowed to depend on a function \(f(i, j, k)\) that depends on the properties of the constituents of the cells (\(i = \) present cell number, \(j = \) connecting cell numbers) and cell wall \((k)\) between cells \(i\) and \(j\). The basic single cell models may be input and output as SBML files using the Cellerator/MathSBML extensions for SBML (Hucka et al., 2003; Shapiro et al., 2004).

**Table 1 | Examples of Cellerator mass action arrow form expansion, from the base from \(A \rightarrow B\).**

| Arrow form* | Expansion |
|-------------|-----------|
| \(e_1 X_1 + e_2 X_2 + \cdots \rightarrow f_1 Y_1 + f_2 Y_2 + \cdots\) | \(e_1 X_1 + e_2 X_2 + \cdots \rightarrow f_1 Y_1 + f_2 Y_2 + \cdots\) |
| \(X \rightarrow Y\) | \(X + E \rightarrow X, E\) |
| \(X \rightarrow Y\) | \(X \rightarrow Y + E\) |
| \(X \rightarrow Y\) | \(X \rightarrow Y\) |
| \(X \rightarrow Y\) | \(X + E \rightarrow X, E\) |
| \(X \rightarrow Y\) | \(X, E \rightarrow Y, E\) |
| \(X \rightarrow Y\) | \(Y, E \rightarrow Y + E\) |

*The complete syntax of an arrow form is \([arrowform, k_1, \ldots]\) where \(k_1, \ldots\) is a sequence of numeric or symbolic rate constants. The subscripts \(e_i\) and \(f_j\) are stoichiometries.
where $\delta$

Fick's law, so that the flux $J$ through any membrane (e.g., in molecules/(cm$^2$-s)) with diffusion constant $D$ (e.g., in cm$^2$/s) is

$$ J = -D \frac{\partial X}{\partial x} \quad (3) $$

Defining the membrane permeability as $\beta = D/\delta$ (e.g., in cm/s), where $\delta$ the membrane (or wall) thickness gives

$$ J = -\beta \frac{\Delta X}{\delta} = -\beta \Delta X \quad (4) $$

where $\Delta X$ is the concentration difference across the membrane. Let cell $i$ have area $A_i$ and depth $d$ (orthogonal to the simulation); the the volume of cell $j$ is $V_j = A_j d$, and the area of the cell wall between cell $i$ and cell $j$ is $\ell_{jk} d$ where $\ell_{jk}$ is the length of the wall between the cells. The flux across wall $k$ into cell $i$ is (for constant area):

$$ J = \left( \frac{d[X_i]}{dt} \times A_i d \right) \times \left( \frac{1}{\ell_{jk} d} \right) = A_i \frac{d[X_i]}{dt} \ell_{jk} \quad (5) $$

Therefore

$$ \frac{d[X_i]}{dt} = \frac{\beta \ell_{jk}}{A_i} \left( [X_j] - [X_i] \right) \quad (6) $$

### Table 2 | Cellerator user defined, regulatory, and enzymatic arrow forms.

| Arrow form$^{a,b}$ | Typical ODE Term$^c$ |
|---------------------|----------------------|
| $\{ e \rightarrow X \Rightarrow f \cdot Y \cdot g(X) \}$ | $[X]' = -e \cdot g(X), \ Y' = f \cdot g(X)$ |
| $\{ X \overset{E}{\leftarrow} Y \cdot Hill \{ v, n, K, a, T \} \}$ | $[Y]' = \frac{v(E) \ (a + T \cdot |X|)^m}{K^a + (a + T \cdot |X|)^n}$ |
| $\{ X \overset{E}{\leftarrow} Y \cdot GRN \{ v, T, n, h \} \}$ | $[Y]' = \frac{v(E)}{1 + \exp(-h - T \cdot |X|)}$ |
| $\{ X \overset{E}{\leftarrow} Y \cdot SSys \{ r, C_+ , C_-, n^+, n^- \} \}$ | $[Y]' = \frac{v(E)}{C_+ \prod_i X_i^m - C_+ \prod_i X_i^n}$ |
| $\{ X \overset{E}{\leftarrow} Y \cdot NHCA \{ \mathbf{T}^+, \mathbf{T}^-, n, m, k \} \}$ | $[Y]' = \frac{v(E)}{k \prod_i (1 + \mathbf{T}^- \cdot (X_i)^k)^m + \prod_i (1 + \mathbf{T}^+ \cdot (X_i)^k)^n}$ |
| $\{ X \rightarrow Y \Rightarrow \text{rational} \{ a, d, m, n \} \}$ | $[Y]' = \frac{a^0 + \text{rest}(a) \cdot |X|^b}{d^0 + \text{rest}(d) \cdot |X|^n}$ |
| $\{ X \rightarrow Y \Rightarrow \text{USER} \{ v, T, n, h, f \} \}$ | $[Y]' = v(f - h - T \cdot |X|)$ |
| $\{ X \rightarrow Y \Rightarrow \text{MM} \{ v, T \} \}$ | $[Y]' = -[X]' = \frac{v(E)}{K + |X|}$ |
| $\{ X \rightarrow Y \Rightarrow \text{MWC} \{ k, n, c, L, k \} \}$ | $[Y]' = -[X]' = k(E) \frac{a(1 + a)^{n-1} + \text{Lac}(1 + a)^{n-1}}{(1 + a)^2 + L (1 + a)^{n-1}}$ |

$^a$The catalyst species $E$ is optional and may be omitted.  
$^b$Boldface quantities may be either scalars or vectors. Vectors enclosed by curly brackets. The notation $v = p^a$ means a vector $v$ with components $v_i = p_i^a$.  
$^c$if the catalyst is omitted replace $[E]$ with 1.  
$^d$in rational, rest(x) is the vector $x$ with its first element removed.  
$^e$In MWC, $a = |X|/K$.

### Table 3 | Additional Cellzilla arrow forms not recognized by Cellerator.

| Arrow form$^a$ | Description |
|-----------------|-------------|
| $\{ X \rightarrow X \cdot \text{Diffusion} \{ P_1, P_0 \} \}$ | Diffusion of $X$ through the tissue. $P_1$ is the permeability of internal cell walls; the optional $P_0$ is the permeability of tissue boundary cell walls. Each may be specified as $f[i,j,k]$ where $i$, $j$, $k$ are cell and wall indices. |
| $\{ X \rightarrow X \cdot \text{Transport} \{ \text{out}, \text{fin} \} \}$ | Controlled transport of $X$ across the cell wall. |
| $\{ X \rightarrow Y \cdot \text{IGRN} \{ v, T, n, h \} \}$ | $\{ Y[j] \}' = \frac{1}{1 + \exp(-h - T \cdot |X[i]|)}$ |
| $\{ \text{cell} \rightarrow \text{cell, Grow} \{ \cdots \} \}$ | Specification of cell growth parameters: Pressure, growth rate, spring constant. |
| $\{ \text{cell} \rightarrow \text{cell, model} \{ \cdots \} \}$ | Specification of cell division model and parameters. |

$^a$Except for the IGRN the arrows here are longer versions of the right-pointing arrows used by the canonical Cellerator mass-action expansion.
Rather than implementing this equation directly, it is implemented as equivalent Cellerator reactions

\[ \frac{dX[i]}{dt} = \delta, \frac{\ell_k}{A_i} \ell(i, j, k), \frac{\ell_k}{A_i} \ell(i, j, k)X[j] \]  

(7)

where \( f \) is the input concentration-dependent permeability. Diffusion is specified to Cellzilla by incorporating arrows of the following form into the model:

\[ \{X \rightarrow X, \text{Diffusion}[f[i, j, k]]\} \]  

(8)

where \( f \) is either a Mathematica pure function or a function that has been defined previously in the simulation.

Action at a distance is technique for describing the way constituents in cell \( i \) can affect constituents in another cell \( j \) without specifying any of the intermediate reactions. We restrict action at a distance to adjacent cells with Cellerator GRN-like reactions written as \( \{X \rightarrow Y, \text{IGRN}[v, \beta, n, h]\} \) which means that constituent \( X_i \) affects constituent \( Y_j \) in neighboring cell \( j \) according to (Mjolsness et al., 1991)

\[ \frac{d[Y_j]}{dt} = \frac{v}{1 + e^{-h - \beta[X_i]}} \]  

(9)

Facilitated membrane transport from cell \( j \) to cell \( i \) is described by the equations

\[ \frac{d[X_i]}{dt} = \frac{\ell_k}{A_i} (f_{\text{out}}[i, j, k] - f_{\text{in}}[i, j, k] - f_{\text{out}}[j, i, k] + f_{\text{in}}[j, i, k]) \]  

(10)

where \( f_{\text{out}}[i, j, k] \) is the positive outward molecular flux from \( i \) through edge \( k \) to \( j \), and \( f_{\text{in}}[i, j, k] \) is the positive inward molecular flux to cell \( i \) through edge \( k \), originating from cell \( j \). In most cases one will only want to specify one of \( f_{\text{out}} \) or \( f_{\text{in}} \). These are implemented internally via the Cellerator reactions

\[ \{X[i] \rightarrow \emptyset, (\ell_k/A_i)(f_{\text{out}}[i, j, k] + f_{\text{in}}[i, j, k])\} \]  

(11)

\[ \emptyset \rightarrow X[i], (\ell_k/A_i)(f_{\text{in}}[i, j, k] + f_{\text{out}}[j, i, k]) \]  

(12)

Transport reactions of this sort are specified to Cellzilla by including arrows of the form

\[ \{X \rightarrow X, \text{Transport}[f_{\text{out}}, f_{\text{in}}]\} \]  

(13)

where the function \( f_{\text{out}} \) should be set to zero if only \( f_{\text{in}} \) is utilized.

**GROWTH MODEL**

Cellzilla implements two types of time-dependent simulations: static, and growing. In static simulations the shape of the tissue and its component parts do not change but its constituents are allowed to vary as described previously. In a growing tissue, the shape of the cells are also allowed to evolve with time. Cell growth is described by associating a Hooke’s law spring potential of the form

\[ V(x) = \frac{1}{2} \sum k_{ij}(\ell_{ij} - \ell_{ij})^2 \]  

(14)

with the edge connecting vertices \( x_i \) and \( x_j \). Here \( \ell_{ij} \) is an equilibrium length assigned to the edge, \( \ell_{ij} \) is the actual length, and \( k_{ij} \) is a constant. When the wall is under compression (so that \( \ell_{ij} > \ell_{ij} \)) there will be a force, acting along the length of the edge, pushing the two vertices apart; when the wall is extended (\( \ell_{ij} < \ell_{ij} \)), the force will tend to pull the vertices toward one-another. The magnitude of this force is equal to the negative gradient of \( V \).

In addition, a pressure \( P_a \) associated with each cell \( a \) is, is applied outward at each vertex. The net force on each vertex is proportional to each wall incident on that vertex and, and is split evenly between the vertices. Then equation of motion for vertex \( x_i \) is then

\[ \frac{dx_i}{dt} = -\sum_j k_{ij} x_i \delta_{ij} + \frac{1}{2} \sum_a P_a n_a \ell_{ij} \]  

(15)

where \( \delta_{ij} \) is a unit vector pointing from \( x_i \) to \( x_j \). The sum in the first term is over all the neighbors \( j \) of vertex \( i \). In the second term, \( n_a \ell_{ij} \) is an outward pointing unit-normal vector from cell \( a \), normal to edge \( \ell_{ij} \) and the sum over all neighbors \( j \) of \( i \) and and over all cells \( a \) in incident at vertex \( i \). The pressure force will cause the springs to extend, simulating cell growth. The resting length is allowed to increase linearly at a rate proportional to the extension beyond resting length,

\[ \frac{dx_i}{dt} = \frac{1}{2} \sum_j k_{ij} x_i \delta_{ij} + \frac{1}{2} \sum_a P_a n_a \ell_{ij} \]  

(16)

where

\[ \Theta(x) = \frac{1}{2}(x + |x|) = \begin{cases} x & \text{if } x \geq 0 \\ 0 & \text{otherwise} \end{cases} \]  

(17)

If the spring is not extended, then growth does not occur. Equations (15) and (16) capture the phenomenological behavior observed in nature by both plant and animal cells. Pressure drives cell expansion; as pressure increases, cells expand more quickly. The growth rate is limited by the spring force. The dynamics of (15) can be solved exactly for a square cell to give

\[ \frac{dL}{L} = k \left( \frac{P}{k} - 2\Delta \right) \]  

(18)

where \( L \) is the cell perimeter and \( \Delta = \sum_i \delta_i \) is the sum of the resting lengths. Consequently, when \( \mu_{ij} = 0 \) in (16), the sigmoidal growth pattern of plant cells is observed [e.g., when extensibility decreases linearly over time and osmotic pressure is held constant, as in Figure 1 of Lockhart (1965)]. Comparing (18) to the Lockhart equation \( \ell' / \ell = \Phi(P - P_E) \) we see that the \( k \) and \( \delta \) are parameters that can tuned to fit effective extensibility \( \Phi \) and yield pressure \( P_E \); for constant \( P \) and \( k \) sigmoidal behavior can be tuned for \( P < 2k \) in square cells. Additionally, allowing \( \mu > 0 \), along
with the spring force produces a more general growth model that is more generally applicable, not just in plant tissue, as the springs can be cut beyond a specified threshold, thereby removing cell-cell interactions (Shapiro and Mjolsness, 2001).

Let \( r \) be the index of the edge connecting vertices \( i \) and \( j \), so that \( \delta_\ell \) and \( \ell_\ell \) are short notations \( \delta_j \) and \( \ell_j \). The spring dynamics of (16) are implemented internally as

\[
\emptyset \to \delta_\ell, \mu_p, q_r, \nu_r (\ell_x - \delta_x)
\]

(19)

where \( \mu \) and \( q \) are the integrals of the cells that about edge \( r \). The pressure force in (15) on vertex \( x \) due to cell \( a \) on the edge connecting vertices \( x \), and \( x_j \) is implemented internally as

\[
\emptyset \to x[i, p], \sum_{\text{Neighbors}(i)} k(r) (x[j, p] - x[i, p]) (1 - \delta_x/\ell_x)
\]

(20)

where \( k(x[i, p]) \) is the \( p \)-th Cartesian component \((p = 1, 2) \) of vertex \( x \), and \( k(r) \) is a spring constant whose value may depend on the properties (e.g., constituents of) the cells abutting edge \( r \). Similarly, the growth of edge \( \delta_\ell \), described by (16) is implemented internally as the Cellerator reactions

\[
\emptyset \to \delta_\ell, \mu_p, q_r, \nu_r (\ell_x - \delta_x)
\]

(20)

where \( \mu \) and \( q \) are the integrals of the cells that about edge \( r \). The pressure force in (15) on vertex \( x \) due to cell \( a \) on the edge connecting vertices \( x \), and \( x_j \) is implemented internally as

\[
\emptyset \to x[i, k], \frac{1}{2} P[a] n[a, i, j, k]
\]

(21)

where \( k = 1, 2 \) (to indicate \( x \) or \( y \) Cartesian component); \( P[a] \) is the pressure in cell \( a \); and \( n[a, i, j, k] \) is the \( k \)-th component of a unit normal vector to edge \( \ell_{ij} \) pointing outwards from cell \( a \).

Cell growth is specified in a Cellzilla model by a reaction of the form

\[
\{\text{cell} \to \text{cell}, \text{Grow}[]{\cdots}\}
\]

(22)

where the arguments to \text{Grow} \[ \cdots \] specify growth parameters such as the dependence of \( k \), \( P \), and \( \mu \) on cell constituents (see Table 3). Chemical concentrations change during growth in each cell occur even in the absence of reactions. If there are \( n \) molecules of \( X \) in volume \( V \) then \( [X]' = (n/V)' = ((Vn' - nV')/V^2 = n'/V - [X]V'/V \). The second term gives the correction in \( [X]' \) due to volume changes.

**CELL DIVISION**

Division occurs when a cell’s area passes a threshold. Upon birth, each cell is assigned a threshold that is distributed normally (with the mean \( \mu \) and standard deviation \( \sigma \) as optional control parameters). Chemical concentrations are distributed equally between the child cells (so that the chemical amounts are proportional to cell area). A single (linear) near cell wall is placed according to one of two user-selectable models: the standard modern interpretation of (Errera, 1888) and a potential model. In the modern interpretation of Errera’s rule, the shortest wall that divides the two cells in half (by area) is chosen. [Technically, this is not Errera’s rule, which only defines the shape of the cell wall, once the endpoints are already known; however, the area-equalization constraint is typically added to provide this boundary condition. (Smith, 2001; Besson and Dumais, 2011; Prusinkiewicz and Runions, 2012)]

In the potential model (Shapiro et al., 2010) a function

\[
V(\theta_1, \theta_2) = \sum_i w_i V_i(\theta_1, \theta_2)
\]

(23)

is minimized over the central angles \( \theta_1 \) and \( \theta_2 \). These give the central angles of the end points of the new wall measured from the cell centroid. Here \( w \) is a weight vector, and \( V_i \) represents each contributor to cell division, where \( i \in \{A, L, e, g\} \), as described in the below.

The area potential \( V_A \) is minimized when the cell divides in half; if the daughter cells have areas \( A_1 \) and \( A_2 \), respectively, then we define

\[
V_A = \left( \frac{A_1 - A_2}{A} \right)^2
\]

(24)

The function is squared to improve computational stability near the minimum (which would otherwise have a non-differentiable corner there). The disadvantage of this potential is that it does not have a unique global minimum, i.e., any line of cell division will that divides the area in half will give a value of zero. Thus, the area potential must be tempered by either an additional potential function (such as the perpendicularity and/or length potential) or an additional heuristic to select the desired minimum value that does not require a unique minimum (e.g., randomly select division direction from amongst all equivalent minima).

The length potential \( V_L \) will be minimized when the new cell wall is most closely aligned with the shortest possible diameter \( d_{\text{min}} \) that this, the shortest line segment dividing the cell passing through the cell center. If \( d \) is the length of the new cell wall, then \( V_L \) is given by

\[
V_L = \frac{(d - d_{\text{min}})^2 + \epsilon_L \Delta^2}{(d + d_{\text{min}})^2}
\]

(25)

where \( \Delta \) is the shortest distance between the new wall and the cell center, and \( \epsilon_L \) is a tunable parameter.

For \( V_e \) and \( V_g \) we define a perpendicularity potential \( V_\perp(v) \) that is minimized when the new wall is perpendicular to particular unit vector \( v \). Let \( W \) be a unit vector parallel to the new wall. Then

\[
V_\perp(v) = v \cdot W + \frac{\epsilon_\perp \Delta}{d}
\]

(26)

where \( \epsilon_\perp \) is a parameter. Letting \( e \) be the direction of maximal cell extension and \( g \) be the direction of maximal cell growth, we then define

\[
V_e = V_\perp(e)
\]

(27)

\[
V_g = V_\perp(g)
\]

(28)

so that \( V_e \) and \( V_g \) are minimized when \( W \) is most nearly perpendicular to the directions of maximal extension and maximal cell growth, respectively. The direction of maximal extension is taken
as the unit eigenvector corresponding to the larger eigenvalue of the covariance matrix $\mathbf{M} = (\mathbf{X}^\top \mathbf{X})/(n-1)$ where $n$ is the number of cell vertices; and $\mathbf{X} = [\mathbf{x} - \mathbf{x}_\text{mean}, \mathbf{y} - \mathbf{y}_\text{mean}]$, where $\mathbf{x}$ and $\mathbf{y}$ are column vectors of cell vertex coordinates $\{(x_1, y_1), \ldots, (x_n, y_n)\}$ and $\mathbf{x}_\text{mean}$ and $\mathbf{y}_\text{mean}$ is their mean. The direction of instantaneous maximal growth is found in the same manner, using the velocities of the vertices. The covariance matrix $\mathbf{M}$ is calculated using the Mathematica function Covariance.

Cell division is specified in a Cellzilla model with the arrow

$$[\text{cell} \rightarrow \text{cell} + \text{cell}.\text{model}][\ldots]$$

where model is either erreraModel or Potential, and the arguments specify the threshold variable, mean, standard deviation, and weight vector (for the potential model). A more generalized version of this notation has been introduced by Yosiphon (2009).

**RESULTS**

**TEMPLATES AND THE BRUSSELATOR**

Cellzilla has variety of shapes that can be used for basic simulations such as rectangular and hexagonal arrays, as well as circular, semicircular and parabolic templates that can be populated with randomly placed Voronoi centers. Alternatively the user can supply a template of his or her own consisting either of cell enters (in which case the walls will be interpolated with a Voronoi algorithm) or cell walls as described previously. Here we present the use of several of these templates (hexagonal, Voronoi, and user-supplied) to implement a common reaction-diffusion system.

The Brusselator (Prigogine and Lefever, 1968) is frequently cited in mathematical modeling because it consists of a system of chemical reactions that in the appropriate parameter regime will maintain sustained oscillations. When combined with diffusion such a system can also be used to establish a wide variety of interesting patterns such as stripes, spirals, and central maxima. The establishment of these different patterns depends on the choice of system geometry, boundary conditions, and parameter values. A diffusible Brusselator is easily implemented in Cellzilla with

$$\{(\emptyset \rightarrow A, a, \beta), [2A + B \rightarrow 3A, c], [A \rightarrow B, b], [A \rightarrow \text{Diffusion}[DA, DB]], [B \rightarrow \text{Diffusion}[DB, DB]]\}$$

where $a, \beta, b, c, DA$ and $DB$ are tunable parameters. As illustrated in Figure 1, the ratio of the diffusion constants will change the number of maxima achieved. In the second row of the figure we see that the geometry is also significant. While the qualitative features of each of the results D-F are identical, their symmetry becomes more and more broken as the symmetry of the template becomes lost. With the exception of the diffusion constants, all other parameters were identical through each of these simulations. We are particularly interested in the parameter set shown in Figure 1A, because it can be used as described in the following section to establish an organizing center for simulation of the WUS/CLV network.

**ESTABLISHMENT OF STEM CELL NICHE**

In Jönsson et al. (2005a,b) we presented a predictive model of feedback interaction between the Wuschel (WUS) and Clavata3 (CLV) signals in the shoot apical meristem. This simplified model was able both to organize the WUS expression domain and to predict the reorganization due to the removal of the CLV signal from the WUS domain as seen in experiments when cells are ablated. This model uses a reaction-diffusion mechanism to induce WUS; the pattern is induced by a Brusselator. The original model relies on a diffusible parameter $Y$ that is produced only in the L1 layer of a slice. We present an implementation in which our slice has the L1 layer omitted, and replace this with a boundary condition in which $Y$ is held fixed, and allowed to diffuse inward. Assuming the Brusselator is implemented by (30), the Cellzilla network for the WUS activator is given by

$$\{[\{Y, A\} \mapsto W, \text{GRM}[\nu, [T_{Y,Y}, T_{W,A}], 1, h, \sigma], [W \mapsto \emptyset, k_w], [Y \mapsto \emptyset, k_y], [A + Y \mapsto Y, d]], [Y \mapsto Y, \text{Diffusion}[D_Y, D_Y]]\}$$

where $\nu$, $T_{Y,Y}$, $T_{W,A}$, $h$, $k_w$, $k_y$ and $D_Y$ are tunable parameters, and the control word $\sigma$ tells Cellerator to replace the usual logistic control function $f(x) = 1/(1 + e^{-x})$ with $f(x) = (1 + 1/\sqrt{1 + x^2})/2$ (in fact, any monotonic increasing saturating function would work). The results illustrated in Figure 2 show that both the original central maximum (in wild type) and dual, smaller maxima result (in the ablation experiment) as modeled.
previously. In addition, we show the steady state distribution of the constituent $Y$ in Figure 2C, illustrating how it forms a ring in the outer cells abutting $L1$ and decreasing inward, as desired.

**GROWTH INDUCED BY ORGANIZING CENTER**

The Brusselator is a useful mathematical/computational artifice for establishing patterns that can be otherwise studied but its biological meaning becomes lost if the variables in the equations do not have biological analogues that are present in the actual tissue. It is more meaningful if the organizing tissue can be established based on specific networks whose constituents have been observed and whose interactions are believed to be present, although this may at times be more computationally intensive. For example, Nikolaev (Nikolaev et al., 2007, 2013) has shown that a combination of reaction-diffusion and feedback in the WUS/CLV network is sufficient to establish a stem cell niche. In a one dimensional dynamic model including cell division, Chickarmane et al. (2012) has shown that negative feedback in the WUS/CLV network is sufficient to establish a stem cell niche; and a second species $V$ that is produced in the $L1$ layer, respectively. Positive feedback of $W$ onto $V$ is provided by a third diffusible species $X$; and the epidermis is impermeable to $U$, $V$ and $W$. Finally, the constitutive degradation of $W$ is slightly enhanced in the $L2$ layer, but occurs everywhere. The network is

\[
\begin{align*}
&\{∅ → U, k_1 TIP[τ]\}, \{U → ∅, k_2\}, \{U → U, \text{Diffusion}[D_U]\}, \\
&\{∅ → V, k_3 L1[τ]\}, \{V → ∅, k_4\}, \{V → V, \text{Diffusion}[D_V]\}, \\
&\{∅ → Z, k_7, k_8 U[τ]\}, \{X → V, \text{GRN}[v_V, T_W, 1, h_V]\}, \\
&\{[U, V, W] → W, \text{GRN}[v_W, T_D, T_V, T_W, 1, h_W]\}, \\
&\{W → ∅, k_6 Z[τ] + k_7 L2[τ]\}, \\
&\{W → X, \text{GRN}[v_X, T_WX, h_X]\}, \{X → ∅, k_5\}. \\
&\{X → X, \text{Diffusion}[D_X]\}, \\
&\{\text{cell} → \text{cell}, \text{Grow}[\text{GrowthRate}[μ, f_μ]], \\
&\text{Pressure}[P, f_p], \text{Spring}[k, f_k]\}, \\
&\{\text{cell} → \text{cell} + \text{Errera}[\text{cell}, μ, σ]\}
\end{align*}
\]

where $k_1, k_2, k_3, k_4, k_5, k_6, k_7, k_8, h_V, h_W, v_V, v_W, T_W, T_D, T_V, D_U, D_V, D_X$ are tunable parameters; $L1[τ], L2[τ],$ and $T1p[τ]$ are built in indicator functions for these cell locations; and the functions $f_p$ and $f_k$ describe pressure and growth feedback; e.g., $P[i] = p_0 + p_1 W[i]$ and $μ[j] = μ_0 + μ_1 (W[i] + W[j])$ (where $p_0, p_1, μ_0$ and $μ_1$ are tunable control parameters). The user would type these function definitions either in the lines before the model or in line as lambda functions in place of the function references in the model. Simulation results are shown in Figure 3. As seen in Figures 3A-C, the stem cell niche is maintained through at least 500 cell divisions.

**DISCUSSION**

We have illustrated that meaningful quantitative results for plant morphodynamics can be obtained using a simple polygonal tissue model coupled with a spring growth equations. In particular, our implementation utilizes and extends an existing arrow-based computational framework that is easy and intuitive to use. This framework is built within a standardized computer algebra system (Mathematica) that is widely available and provides access to a wide selection of analytical tools. In addition we believe that our framework is generalizable and extensible to the wider world or rule-based systems. While more detailed particle-based or molecular dynamics frameworks will certainly produce more accurate frameworks their ultimate extensibility is limited by CPU availability and time constraints. Our implementation provides a useful platform for rapid model development and testing that can easily be transformed to one of the more detailed frameworks, if desired, once suitable results are obtained.

Cellzilla can be used for 2D simulations of plant tissues at the multicellular level. Because models can be rapidly built and

![FIGURE 2 | Results of Wuschel simulation using Cellzilla. Species concentration is illustrated at the end of the simulation time course. The concentration of Wuschel is shown in blue, and protein Y is shown in green. Darker colors indicate higher concentrations, and white indicates a zero concentration. Steady state concentration of (A) Wuschel in wild-type simulation; (B) Wuschel in ablated meristem; (C) signal protein Y in ablated meristem. Parameters: $v = 0.1$, $T_W = 0.2$, $T_X = 0.5$, $b = 0$, $k_y = 0.1$, $k_x = 0.1$, $b = 0.5$, $D_T = 2$, $D_X = 1.5$, $D_V = 15$.](image)

![FIGURE 3 | Maintenance of organizing center during growth simulation. (A) Initial distribution of $W$; (B) Distribution after 250 cell divisions; and (C) Distribution after 500 cell divisions. (D) Cell lineages; each color shows a different clonal population corresponding to all descendants of a particular cell in (A). Parameters: $k_1 = 2$, $k_2 = 0.2$, $k_3 = 1$, $k_4 = 0.25$, $k_5 = 1$, $k_6 = 0.05$, $k_7 = 1.5$, $k_8 = 0.1$, $D_U = 10$, $D_V = 0.5$, $D_X = 0.5$, $V_4 = 1$, $V_5 = 1$, $V_6 = 0$, $h_V = 0$, $h_W = 0$, $V_4 = 0$, $T_W = 4$, $T_X = 4$, $T_V = 22.5$, $T_W = 25$, $T_V = 27.5$, $p_0 = 0.001$, $p_1 = 0.004$, $μ_0 = 5 \times 10^{-6}$, $μ_1 = .004$.](image)
tested it allows one to quickly test developmental models both in steady-state and during growth. However, several improvements are planned for future versions. In the current version, only simple transport and diffusion are considered; however, in many cell types, not just plant cells, osmotic and electrical gradients are significant. We plan to implement rules to incorporate both features in future releases. Furthermore, with the addition of osmotic models and pressure gradients we plan to add additional plant-specific growth models (Lockhart, 1965; Ortega, 1985; Geitmann and Ortega, 2009). These can be optionally used in place of the phenomenologically-based spring-based growth model. Additions to the cell division model will be included. For example, the constraints used in the Errera implementation of linear walls and equal areas can be relaxed to quadratic and circular arcs, and the areas can be randomized in a cloud about equality.

A three-dimensional model is also planned, although we expect that this will significantly improve computational demands. To avoid mathematically unstable solutions in three dimensions the spring model will need to be modified or replaced, most likely with an elastic dynamics model incorporating pressure and stress tensors. An alternative is the use of triangular springs (Delingette, 2008).

DATA SHARING
All of the software described here is open source Mathematica (GPL license) and freely downloadable from launchpad at https://launchpad.net/cellerator. The software is fully documented and all examples are available at the project website https://www.cellzilla.info.

ACKNOWLEDGMENTS
We would like to thank Sergey Nikolaev for valuable suggestions. This work was supported by grants from the Beckman Institute, the Division of Biology, and the Provost’s Office at Caltech; a gift from Peter Cross to Caltech; by the Howard Hughes Medical Institute and the Gordon and Betty Moore Foundation (through Grant GBMF3406) to Elliot M. Meyerowitz; and by NIH Grant R01 GM086883 to Eric D. Mjolsness at UCI.

REFERENCES
Andrews, S. (2012). “Spatial and stochastic cellular modeling with the smoldyn simulator,” in Bacterial Molecular Networks: Methods and Protocols: Methods in Molecular Biology, eds J. van Helden, A. Touissant, and D. Thieffry (New York, NY: Springer), 519–540.

Besson, S., and Dumais, J. (2011). Universal rule for the symmetric division of plant cells. Proc. Natl. Acad. Sci. U.S.A. 108, 6294–6299. doi: 10.1073/pnas.1011866108

Chickarmane, V., Gordon, V., Tarr, S., Besson, S., and Dumais, J. (2011). Additional plant-specific growth models (Lockhart, 1965; Ortega, 1985; Geitmann and Ortega, 2009). These can be optionally used in place of the phenomenologically-based spring-based growth model. Additions to the cell division model will be included. For example, the constraints used in the Errera implementation of linear walls and equal areas can be relaxed to quadratic and circular arcs, and the areas can be randomized in a cloud about equality.

A three-dimensional model is also planned, although we expect that this will significantly improve computational demands. To avoid mathematically unstable solutions in three dimensions the spring model will need to be modified or replaced, most likely with an elastic dynamics model incorporating pressure and stress tensors. An alternative is the use of triangular springs (Delingette, 2008).

DATA SHARING
All of the software described here is open source Mathematica (GPL license) and freely downloadable from launchpad at https://launchpad.net/cellerator. The software is fully documented and all examples are available at the project website https://www.cellzilla.info.

ACKNOWLEDGMENTS
We would like to thank Sergey Nikolaev for valuable suggestions. This work was supported by grants from the Beckman Institute, the Division of Biology, and the Provost’s Office at Caltech; a gift from Peter Cross to Caltech; by the Howard Hughes Medical Institute and the Gordon and Betty Moore Foundation (through Grant GBMF3406) to Elliot M. Meyerowitz; and by NIH Grant R01 GM086883 to Eric D. Mjolsness at UCI.
expansion. *Plant Physiol.* 79, 318–320. doi: 10.1104/pp.79.1.318

Plimpton, S. J., and Slepoy, A. (2005). Microbial cell modeling via reacting diffusing particles. *J. Phys.* 16, 305–309. doi: 10.1088/1742-6596/16/1/042

Pradal, C., Dufour-Kowalski, S., Boudon, F., Fournier, C., and Godin, C. (2008). Openalea: a visual programming and component-based software platform for plant modeling. *Funct. Plant Biol.* 35, 751–760. doi: 10.1071/FP08084

Prigogine, L., and Lefever, R. (1968). Symmetry breaking instabilities in dissipative systems. II. *J. Chem. Phys.* 48, 1695–1700 doi: 10.1063/1.1668896

Prusinkiewicz, P., and Lindenmayer, A. (1990). *Algorithmic Beauty of Plants.* New York, NY: Springer Verlag.

Prusinkiewicz, P., and Runions, A. (2012). Computational models of plant development and form. *New Phytol.* 193, 549–569. doi: 10.1111/j.1469-8137.2011.04009.x

Rudge, T., and Haseloff, J. (2005). A computational model of morphogenesis in plants. *Adv. Artif. Life* 3630, 78–87.

Sahlin, P., and Johnsson, H. (2010). A modeling study of how cell division affects properties of epithelial tissues under isotropic growth. *PLoS ONE* 5:e11750. doi: 10.1371/journal.pone.0011750

Shapiro, B. E., Heisler, M., Tobin, C., Cunha, A., Davis, A., Mjolsness, E. D., et al. (2010). “Using geometric markers to predict the cell division plane in meristem cells,” in *Proceedings of the 6th International Workshop on Functional-Structural Plant Models*, eds T. DeJong and D. Da Silva (Davis, CA: University of California), 144–146.

Shapiro, B. E., Hucka, M., Finney, A., Doyle, J. (2004). MathSBML: a package for manipulating SBML based biological models. *Bioinformatics* 20, 2829–2831. doi: 10.1093/bioinformatics/bth271

Shapiro, B. E., Levchenko, A., Meyerowitz, E. M., Wold, B. I., and Mjolsness, E. D. (2003). Cellerator: extending a computer algebra system to include biochemical arrows for signal transduction simulations. *Bioinformatics* 19, 677–678. doi: 10.1093/bioinformatics/btg042

Shapiro, B. E., and Mjolsness, E. D. (2001). “Developmental simulations with cellerator,” in *International Conference on Systems Biology* (Pasadena, LA).

Smith, L. G. (2001). *Plant Cell Division: Building Walls in the Right Places.* Nat. Rev. 2, 33–39. doi: 10.1038/35048050

Sneddon, M. W., Faeder, J. R., Emonet, T. (2011). Efficient modeling, simulation and coarse graining of biological complexity using nfsim. *Nat. Methods* 8 177–183. doi: 10.1038/nmeth.1546

Stiles, J. R., and Bartol, T. M. (2001). “Monte carlo methods for simulating realistic synaptic microphysiology using MCell,” in *Computational Neuroscience: Realistic Modeling for experimentalists*, ed E. De Schutter (Boca Raton, FL: CRC Press), 87–127.

Yang, C.-R., Shapiro, B. E., Mjolsness, E. D., and Hatfield, G. W. (2003). An enzyme mechanism language for the mathematical modeling of metabolic pathways. *Bioinformatics* 21, 774–780. doi: 10.1093/bioinformatics/bti068

Yosiphon, G. (2009). *Stochastic Parameterized Grammars: Formalization, Inference, and Modeling Applications.* PhD Thesis, UC Irvine.

Conflict of Interest Statement: The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.