A Framework for Three-Dimensional Simulation of Morphogenesis

Trevor M. Cickovski, Chengbang Huang, Rajiv Chaturvedi, Tilmann Glimm, H. George E. Hentschel, Mark S. Alber, James A. Glazier, Stuart A. Newman, and Jesús A. Izaguirre

Abstract—We present COMPUCELL3D, a software framework for three-dimensional simulation of morphogenesis in different organisms. COMPUCELL3D employs biologically relevant models for cell clustering, growth, and interaction with chemical fields. COMPUCELL3D uses design patterns for speed, efficient memory management, extensibility, and flexibility to allow an almost unlimited variety of simulations. We have verified COMPUCELL3D by building a model of growth and skeletal pattern formation in the avian (chicken) limb bud. Binaries and source code are available, along with documentation and input files for sample simulations, at http://compucell.sourceforge.net.

Index Terms—Cellular Potts Model (CPM), reaction-diffusion, cellular automata, morphogenesis, Extensible Markup Language (XML).

1 INTRODUCTION

THIS paper presents COMPUCELL3D, a three-dimensional (*3D*), cell-centered, multiscale [17] framework for modeling morphogenesis. Morphogenesis is the structural development of an organism and its organs, involving cell differentiation, growth and migration, bulk changes in tissue shape, and the secretion, resorption and diffusion of extracellular materials (e.g., proteins and tissue polysaccharides). Cell interactions via secreted and membrane-bound chemicals generate biologically significant patterning instabilities that we can describe mathematically and implement computationally [32], [35], [38], [46], [50], [59], [71], [96], allowing us to model morphogenesis [37]. For this mathematical and computational modeling of development, the cell provides a more useful level of abstraction than the subcellular level. The cell-centered approach studies how the behaviors of individual cells collectively influence behavior at higher levels, such as tissues and organisms. For example, in some cases, such as the function of an inductive signal, what happens "behind the scenes" is irrelevant to macroscopic model behaviors, since all that matters is that cells respond to the signal correctly. Meinhardt [60] recently stated that "The role of the cell as a module of development can hardly be overestimated" and Von Dassow and Meir [90] argue that the closest thing to data hiding in nature is the cell. Cell-level modeling helps tackle complexity and makes modeling feasible. For example, a typical cell can contain roughly $10^5 \sim 10^6$ gene products. Estimating potential interactions between all of these products would be needlessly complicated and extremely expensive in time and space when run on a computer, even a highly parallel machine. By treating cells phenomenologically and ignoring intracellular behaviors, we can reduce multiple complex interactions to a small set of behaviors such as movement, division, death, differentiation, shape change, force exertion, chemical and electrical-charge secretion and absorption, and surface property distribution changes. At the other extreme, macroscopic models like *Physiome* [10] which operate at the level of tissues, while highly efficient, cannot reproduce some experimental observations that arise from finer-grained behaviors at the cell level.

Merks and Glazier [61] provide a solid set of steps for building a cell-centered model:

1. Infer individual cell behaviors from biological experiments. We can obtain some behaviors of specific cells from the literature. Sometimes, we need additional experimentation for specific numerical information.
2. Outline a conceptual model of the cell behaviors.
3. Translate this abstraction into a mathematical model.
4. Implement the mathematical model computationally into a *simulation*. Although the conceptual model may apply to a *single cell*, computationally, we can combine multiple single-cell models to determine if the model suffices to describe the tissue-level interactions and functionality that we found experimentally.
5. If the simulation results do not match experimental results, return to the experiment to identify inaccuracies or missing information or parameters in the

- *T.M. Cickovski and C. Huang are with the Laboratory for Computational Life Sciences, Department of Computer Science and Engineering, University of Notre Dame, 325 Cushing Hall, Notre Dame, IN 46556. E-mail: {tcickovs, chuang}@nd.edu.*
- *R. Chaturvedi is with the Department of Computer Science and Engineering, University of Notre Dame, 384 Fitzpatrick Hall, Notre Dame, IN 46556.*
- *T. Glimm and H.G.E. Hentschel are with the Department of Physics, Emory University, 400 Dowman Drive, Atlanta, GA 30322. E-mail: tglimm@emory.edu, phshgeh@physics.emory.edu.*
- *M. Alber is with the Department of Mathematics, University of Notre Dame, 255 Hurley Building, Notre Dame, IN 46556. E-mail: malber@nd.edu.*
- *J.A. Glazier is with the Department of Physics, Indiana University, Swain Hall West 159, Bloomington, IN 47405. E-mail: glazier@indiana.edu.*
- *S.A. Newman is with the Basic Science Building, New York Medical College, Valhalla, NY 10595. E-mail: newman@nymc.edu.*
- *J.A. Izaguirre is with the Department of Computer Science and Engineering, University of Notre Dame, 326C Cushing Hall, Notre Dame, IN 46556. E-mail: izaguirr@nd.edu.*

Manuscript received 2 Aug. 2004; revised 25 Jan. 2005; accepted 25 Apr. 2005; published online 31 Aug. 2005.
For information on obtaining reprints of this article, please send e-mail to: tcbb@computer.org, and reference IEEECS Log Number TCBB-0083-0804.

model, e.g., using high-throughput parameter studies. Conversely, parameter sweeps of the computational model can identify possible behaviors to search for experimentally. Additional inputs can refine the model: For example, we can measure the relative strength of cell adhesion between different cell types from surface tension experiments [30]; knock-out experiments can validate the role of identified signals; and modeling the relevant gene networks can complete the picture the model offers.

6. Once the simulation results match experimental results, study further test cases by, e.g., removing a parameter and recording the changes in behavior. If we observe accurate behavior at the tissue level even though we have removed certain elements of the model, we have a smaller set of behaviors that can still accurately simulate experimental results. We can eventually obtain a minimal necessary set of single-cell behaviors through repeated testing.
7. Once we have established this minimal set of behaviors, we determine which gene networks drive this set of behaviors, and how they operate. At this point, we can ask questions about the factors which result in abnormal growth—as we have done for a simpler model of chondrogenesis which we have experimentally validated using cell cultures (cf. Section 6).

The Cellular Potts Model (*CPM*) provides a well-defined, cell-centered framework for simulations of morphogenesis [35]. The CPM is a grid-based, stochastic model designed to accurately simulate cell interactions and movement. It can reproduce cell membrane fluctuation in a way that matches experiments on cell dynamics, even though it neglects cellular substructures like the cytoskeleton. It models mesenchymal cells, which are relatively isotropic, without requiring further extensions. Extra terms in the CPM Hamiltonian allow the CPM to model highly polar cells, e.g., [94], [93], [62]. Some of the many studies using the CPM include Mombach and Glazier's [67] study of chicken retinal cells and Marée's [55], [57], [58], [56] study of *Dictyostelium discoideum*. Many of these studies include direct, quantitative validation of the CPM results against experiments. The idea that cell-cell interactions depend on surface-tension forces and their analogues goes back well before Thompson, who was one of the first to attempt a semiquantitative comparison [84]. Steinberg then developed this idea into his Differential Adhesion Hypothesis [80], [81], [30], [26]. Experimental studies on cell sorting *in vitro* [30], [26] and, more recently, *in vivo* [33], [34], [36], support this approach to tissue morphogenesis based on consideration of surface forces akin to those that govern the behavior of bubbles and foams.

Recent modifications to the CPM extend its biological accuracy. Ouchi et al. [73] modified the Hamiltonian to use negative surface energies, constrained surface areas, and a spin-flip energy threshold to improve the correspondence to reality. The modified model correctly predicted several dynamical behaviors of cells which the original CPM did not, including the hierarchy of diffusion constants. Mombach et al. [68] studied the influence of surface tension and size on the rounding of chick embryonic cell aggregates and CPM simulations. The results showed exponential relaxation in both cases, which studies using 2D *Hydra* cell aggregates verified [76], [66], [88]. The relaxation time decreased with higher surface tension as expected from hydrodynamic laws [12], [11]. However, it increased faster than linearly with aggregate size. The results provided additional validation of the CPM in nonequilibrium situations. Merks and Glazier [61] surveyed the validity of multiscale computational models in developmental biology. The cell-centered approach that they favor uses phenomenological models of single-cell behaviors such as differentiation in response to signals, cell-cell adhesion, cell-extracellular matrix interactions, and chemotactic and haptotactic responses to gradients, to build a biological model for tissue and organ-level patterns and functions. This approach is the essence of the CPM. Merks and Glazier [61] also used the CPM as a paradigm for cell-centered modeling of cell rearrangement driven by differential adhesion [95] during morphogenesis, and argued for its advantages over other cell-centered models in accounting for cell shapes, distinguishing between cell adhesion and attraction, and ease of integrating different software frameworks into the model.

2 CompuCell3D

CompuCell3D takes a hybrid approach to modeling morphogenesis [16], combining discrete cellular-automaton and continuum methods. We implement the CPM as a cellular automaton [4] governing cell interactions, along with reaction-diffusion (*RD*) equation solvers to establish surrounding chemical gradients. Domain growth is another key factor; e.g., researchers have studied the effect of uniform [53] and spatially nonuniform [22] growth on biological development. Dillon and Othmer implemented a domain-growth model of the shaping of the developing vertebrate limb using a continuum approach [25]. CompuCell3D includes a 3D density-dependent growth algorithm. Navier-Stokes, reaction-advection-diffusion (*RAD*) equation solvers, have also reproduced two-dimensional (*2D*) patterns and simulated growth. Coarse RAD models are fast. However, solving the RAD equations in detail is difficult because advection and moving boundaries can cause numerical instabilities. We avoid solving RAD equations by using a growth algorithm which depends only on the local cell density in the Potts lattice, and enforces domain growth when the density exceeds a threshold. Dan et al. [23] have recently developed a fast RAD solver that runs entirely within the CPM framework and which doubles as a Navier-Stokes solver for low Reynolds-Number flows.

CompuCell3D can simulate morphogenesis in both multicellular and unicellular organisms such as Myxobacteria [5]. Myxobacteria simulations require the ability to emulate the cell polarity of unicellular organisms by modeling single cells as multiple lattice domains with varying adhesivity. To study the role of polarity in cell aggregation [3], [5], [51], we have implemented these approaches within the CPM framework of CompuCell3D. Our CPM simulations of Myxobacteria implement polarity

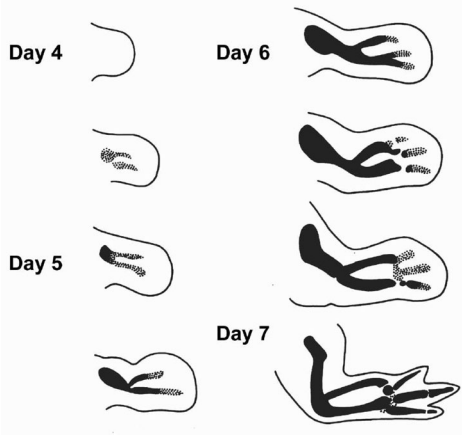

Fig. 1. Developmental timeline of chick-limb skeletal patterning. Drawings show transverse sections of wing buds. For all panels, proximal is left, distal right, anterior up, and posterior down. From [71], with modifications.

within single cells, allow migration in a direction dependent upon the cell's orientation, and preserve a rod-shaped cell using a cylindrical constraint. Our simulations successfully reproduce the directional movement and shapes of experimental rod-like bacteria. The cellular-automaton approach Hösel and Liebscher proposed for modeling nutrient supply to biofilms [43] employs a Gibbsian probability to model cell interactions and uses an energy function with a realization probability inversely proportional to the configuration energy, which would also translate easily into the COMPUCELL3D framework.

In order to illustrate COMPUCELL3D, we have built a 3D model of skeletal patterning in the experimentally well-studied avian limb. Because the skeletal pattern is first established as cartilage before being replaced by bone, we call this patterning *chondrogenic* (cartilage-forming). During embryonic development, the vertebrate limb progressively generates a sequence of increasing numbers of cartilage elements *proximo-distally*. That is, the first elements to form are those closest (*proximal* to) to the body wall, and the last are those farthest from (*distal* to) the body. In a forelimb, this sequence begins with the humerus, followed by the radius and ulna, then the carpals and metacarpals and, finally, the digits. Although the bones at any given proximo-distal level are more similar to each other than to those farther up or down the limb, they also differ significantly in the *antero-posterior* direction, that is, the direction defined by the thumb to the little finger. Fig. 1 shows a developmental timeline of patterning in the avian limb bud, viewed with the proximo-distal axis running from left to right, and the antero-posterior axis running from top to bottom. The *dorso-ventral* axis of the limb, defined by the axis from the back of the hand to the palm, points out of the page in this representation.

To construct our cell-based model in COMPUCELL3D, we perform the following steps: First, we obtain cell-behavior data from the scientific literature and experiments. In our verification simulation, we assume an irregular mesenchymal cell morphology and obtain mitosis rates from [54], define an activator signal which we identify with TGF-β [63], a putative inhibitor [59], [64], [85], and an ECM component which the cell secretes, which we identify with fibronectin [38], [50]. We assume that the cell undergoes haptotaxis (i.e., directed, contact-dependent movement) in response to fibronectin concentration gradients. From chondrogenic experiments [71], we define two types of cells into which precartilage mesenchymal cells can differentiate: "condensing" (i.e., forming tight cell aggregates) and "noncondensing." These biological models translate into three computational models: a set of RD equations, with initial and boundary conditions coming from experimentally-observed cell densities and concentrations; a CPM model incorporating all the single-cell behaviors we identified above (this single-cell model is merely descriptive, but becomes useful when the cell interactions produce emergent patterns that resemble those in the real chick limb bud) and a Cell-Type-Map that models differentiation of cells.

Hentschel et al.'s [38] recent model of chondrogenesis provides an experimentally-motivated "bare-bones" mechanism for the major features of limb skeletal patterning. In particular, the model proposes a core network of cell, biochemical, and genetic interactions known to be involved in producing precartilage condensations in limb-cell cultures and in developing limbs (see [38] and [50] for a detailed description and experimental cell biology background). The core mechanism has RD form and governs the spatiotemporal evolution of the condensation pattern [2].

A model of this sort must necessarily leave out many of the scores of biochemical and genetic interactions that contribute to generating a real limb. The biology of precartilage condensation, however, dictates that we must include cell adhesion and certain signals which experiments show to be indispensable. Whether or not such a minimal set of cell biological interactions can form limb-like skeletal patterns without additional molecular apparatus is a nontrivial question that only modeling and simulation can address. Hentschel et al. [38] answered this question affirmatively for a two-dimensional model limb which represents cell density as a continuous variable. Kiskowski et al. [50], representing cells as autonomous agents and using a similar, but simplified, set of biological interactions, showed that regular patterns of cell condensation qualitatively and quantitatively like those seen in limb-cell cultures will arise for a narrow but robust set of parameter values.

The dynamics of limb growth define in a natural fashion three experimentally-determined zones in the developing limb: an *apical zone* in which a reserve of cells remains unpatterned, an *active zone* in which cells rearrange and condense, and a *frozen zone* in which condensing cells differentiate into cartilage and cease rearranging (see Section 6 and [38]). Our model explicitly encodes neither the zonal organization of the developing limb nor the spatiotemporal development of the skeleton. They are, instead, emergent biological properties (resulting from the symmetry assumptions of [38]) of both the 2D continuum, bare-bones mechanism, and its more realistic embodiment in a COMPUCELL3D simulation (see Section 6).

We establish an exterior chemical gradient by solving the RD equations to obtain the concentration field of a diffusible *activator* molecule, which we identify with the positive autoregulatory growth factor TGF-β [63]. An *inhibitor* molecule suppresses the production, or downstream effects,

of the activator [59], [64], [85]. We assume that the cells respond to the activator by producing a secreted molecule, *fibronectin*, to which they adhere (see [38] and [50] for additional details).

The RD equations are:

$$\begin{aligned}\frac{\partial c_a}{\partial t} &= \gamma[(J_0 + J_a(c_a)\beta(c_a))R_0 - k_a c_a c_i] + b_a(c_a - c_{as})^3 \\ &\quad + \left(d_{ax}\frac{\partial^2 c_a}{\partial x^2} + d_{ay}\frac{\partial^2 c_a}{\partial y^2} + d_{az}\frac{\partial^2 c_a}{\partial z^2}\right), \\ \frac{\partial c_i}{\partial t} &= \gamma[J_i(c_a)\beta(c_a)R_0 - k_i c_a c_i] + b_i(c_i - c_{is})^3 \\ &\quad + D\left(d_{ix}\frac{\partial^2 c_i}{\partial x^2} + d_{iy}\frac{\partial^2 c_i}{\partial y^2} + d_{iz}\frac{\partial^2 c_i}{\partial z^2}\right),\end{aligned} \tag{1}$$

where c_a and c_i represent the respective concentrations of activator and inhibitor, c_{as} and c_{is} are the spatially-homogeneous steady states for the activator and inhibitor concentrations, and R_0 denotes the average cell density.

This system represents a modified form of the equations of Hentschel et al. [38] which we implemented in COMPUCELL3D. We kept only the two dominant equations and added two terms $b_a(c_a - c_{as})^3$ and $b_i(c_i - c_{is})^3$ to enforce stability [29], [1]. As noted above, these equations represent known biological interactions in the chick limb. We used them to generate the chemical field in our verification simulations. The emergence of the sequence of bone structures resulted from changes in the domain geometry as well as in the reaction kinetics.

We first describe the CPM in detail, along with ways in which it can model biological mechanisms such as cell adhesion, cell growth and division, reaction-diffusion, chemotaxis and haptotaxis, and cell type and state. We then describe how COMPUCELL3D addresses the inherent issues present in computational modeling of morphogenesis. Next, we describe various software techniques which we used in the design of COMPUCELL3D. These include polymorphism to make the framework extensible and user-friendly, as well as various computational techniques such as *offset-neighbor evaluation* which uses a lazy calculation of neighbor pixels in a grid for a four-fold increase in computational speed and a 10-fold reduction in memory consumption compared to standard alternatives. Finally, we present our 3D avian limb-bud simulation, which includes cell division. We provide sample input files for this simulation, along with instructions for running COMPUCELL3D, on the COMPUCELL Web site [21].

COMPUCELL3D can also work in tandem with other existing software frameworks treating subcellular and supercellular phenomena. For example, BioSPICE [13], [7] models dynamic, cellular network functions. BioSPICE can clarify complex intracellular biochemical networks [52] to simulate cell division, circadian rhythms, bacterial sporulation, and gene transcription [24]. BioSPICE simulations of known regulation and signaling networks can either control COMPUCELL3D cell parameters directly or generate simplified ODEs and state-change-map models of cell behaviors which the user can implement in the Cell-Type-Map module of COMPUCELL3D. CellO [15] is an object-oriented tool that can model apoptosis, induction, differentiation, and the cell cycle. In contrast to the CPM, CellO uses a grid-independent method to model cell motion, using attractive forces to model cell adhesion and repellent forces to model cell elasticity. We can cross-validate the results of COMPUCELL3D by implementing identical models using CellO's grid-independent methods. NEURON [69], [39] provides a simulation environment for neuron modeling, specifically supporting complex cell membrane properties. NEURON can treat large groups of cells and connections and possesses a GUI with a CellBuilder that can create new models or modify existing models, a potentially useful tool to interface with COMPUCELL3D. E-Cell [27], [83] is an object-oriented software suite for analysis of large-scale biological interactions, including biological cells. Researchers are currently developing E-Cell 3 as a platform for integration of multiple algorithms such as reaction-diffusion, cellular automata, and Gillespie's algorithm [31] for stochastic simulation of coupled chemical reactions. E-Cell 3 will be another useful tool for cross-validation of COMPUCELL3D. Finally, Virtual Cell [89], [78] can model cellular physiology by allowing a user to define both biological models which include chemical species and reactions, subcellular structures, and cell geometry, and mathematical models via a general-purpose solver for steady and unsteady solutions of algebraic equations, including partial differential equations (*PDE*s) and ordinary differential equations (*ODE*s). Virtual-Cell models can set up the initial state for a COMPUCELL3D simulation including cell positions and structure, and we can use Virtual-Cell equation solvers to generate external chemical gradients.

3 A MODEL FOR MORPHOGENESIS

The CPM uses a lattice to describe cells, and associates an integer index with each lattice site (*voxel*) to identify the spatial extent and location of each cell at any instant. The index value at a lattice site is σ if the site lies in cell σ. *Domains* in the lattice (the collection of lattice sites with the same index) represent cells. A cell is thus a set of discrete components that can rearrange to produce cell shape changes and motion. The CPM follows the principle of energy minimization, with the configuration of cells gradually rearranging to reduce the generalized pattern energy.

Fig. 2 shows three 2D cells and their extracellular matrix (*ECM*), which require four distinct indices. It also demonstrates the scheme for determining pixel neighbors and their levels, a key part of the CPM.

3.1 Principle of Energy Minimization

In the CPM an effective energy, E, determines cell interactions, motion under cytoskeletal fluctuations, response to external chemical stimuli, differentiation, and division. The effective energy contains true energies (e.g., cell-cell adhesion) and terms that mimic energies (e.g., the response of a cell to a chemotactic gradient). A pattern evolves under strong damping to reduce its energy. Such an approach is valid, because, in the high-damping limit, motion is Aristotelian rather than Newtonian, i.e.:

$$\vec{v} = \mu\vec{F} = \mu\vec{\nabla}E. \tag{2}$$

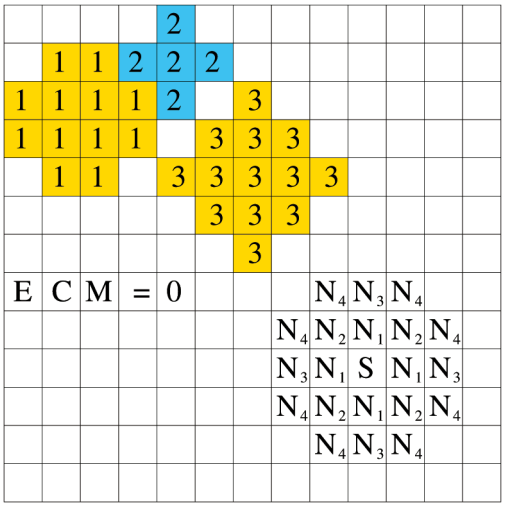

Fig. 2. The CPM grid showing cells and ECM. The shading denotes the cell type. Different cells (for example, cells 1 and 3) may have the same type. A site S connects up to fourth-neighbor pixels $(N_1, \ldots, N_4)$.

In this case, we can describe any biological mechanism where the response of a cell is a velocity proportional to the gradient of a scalar quantity simply by introducing that scalar quantity as an effective energy. For example, in chemotaxis, the simplest assumption is that the velocity of a cell is proportional to the gradient of the chemical concentration, $E_{chemical} = \mu\vec{\nabla}C$. In this case, the chemical concentration times an effective chemical potential (equal to the mobility μ) has the form of a potential energy and we can include the chemotaxis term as an effective energy. Of course, if we wanted to account for saturation of cell receptors for large chemical concentrations and for limitations in cell velocity, we could use a Michaelis-Menten or Hill form for our chemical potential [62], [77]:

$$E' = E - \sum_i \chi \, \frac{c(\vec{x,t})}{s \; c(\vec{x,t}) + 1}(1 - \delta_{\sigma_i,\sigma_j}). \tag{3}$$

Upadhyaya [87] and Marée [56] have justified these aspects of the CPM quantitatively, reproducing the behavior of different kinds of cell aggregates. The dynamics favor connected domains of lattice sites with the same index.

In mixtures of liquid droplets, thermal fluctuations of the droplet surfaces cause diffusion, or *Brownian motion*, leading to gradual pattern rearrangement as the cells diffuse in their environment. The effective energy biases the movement, applying small forces which result in eventual energy minimization. The simplest phenomenological assumption is that an *effective temperature*, T, drives cell-membrane fluctuations. T defines the size of the typical fluctuation. We implement fluctuations using the Metropolis algorithm for Monte-Carlo Boltzmann dynamics. Beysens et al. [12], [11] have conducted experiments that justify this simplification. If a proposed change in lattice configuration (a change in the indices associated with the voxels of the lattice) produces a change in effective energy ΔE, we use the *acceptance function*:

$$P(\Delta E) = \begin{cases} 1, & \text{if } \Delta E \leq 0, \\ e^{-\Delta E/kT}, & \text{if } \Delta E > 0, \end{cases} \tag{4}$$

where k is a constant converting T into units of energy. We can also account explicitly for energy dissipation in making and breaking contacts by employing an extra dissipation. For more detailed information on acceptance function, see [92].

Three possible objections to the CPM approach are that it does not describe the cytoskeleton and that it does not handle force transduction or dissipation in a transparent manner. These objections do indeed point to omissions in the simplest versions of the CPM, which is why we have made an extra effort to validate CPM results quantitatively against relevant experiments.

In answer to the first objection, we note that in these simulations we are separating the emergent behaviors of aggregates of cells, which we study, from the internal mechanisms which give rise to those behaviors. We think that this approach optimizes trade offs between model complexity, execution time, and the amount of available biological knowledge. For our present purposes, the cytoskeleton appears indirectly through several cell behaviors: Cells diffuse randomly in their environment, which models the effects of membrane ruffling; they undergo chemotaxis and haptotaxis, which models the formation of leading edges; they undergo mitosis; and they can develop variable shapes and distributions of cell adhesion molecules on their surfaces. The mesenchymal cells we simulate in our limb-bud example are essentially baglike and nonpolar. It is straightforward to introduce more biological detail as it becomes available. What is striking is that, even with radical simplifications of cell behaviors, simulated cell aggregates still reproduce, quantitatively, many complex observed phenomena, like the entire life cycle of *Dictyostelium*. The principle of Occam's razor reminds us to build and understand minimal models before introducing additional complexity. We should avoid introducing mechanisms that bring with them large numbers of parameters (whose values are usually undetermined experimentally), but which are not qualitatively necessary to the behaviors we are studying. Complexity theory tells us that the behaviors of large aggregates depend on the general features of the agent interactions and are relatively insensitive to the details of those interactions. As cytoskeletal simulations develop, we could easily interface a cytoskeletal model like that of Mogilner to the CPM [65].

In answer to the second objection, we note that in none of the cases we treat here is long-range force transduction significant. In the quasifluid environment of soft tissues, we have established that the flows the CPM predicts are realistic [46], [47], [9], [45]. For the purposes of developmental modeling, because movements are so slow and deformations so small, in most cases the exertion of force against the fixed underlying lattice is an adequate proxy for force exertion against the ECM. We are currently developing a methodology for simulating rigid-body motion within the CPM framework. COMPUCELL3D supplies hooks so that the user can interface it to any standard finite-element package if highly-accurate rigid-body or force-transduction simulations are desired.

The issue of dissipation we can answer partially by using the modified acceptance function we have described above. In the highly viscous regime we work in, the major effect of changing dissipation rates is to change the relationship of a Monte Carlo Step (see below) to time. In addition, while groups using magnetic and optical tweezers have made some measurements of cytoplasmic and cellular dissipation, we almost never know the actual dissipation rates significant *in vivo*. As we noted above, one of our guiding principles is to introduce as few parameters with unknown values as possible. The RAD solver that we have developed and will soon release in an update to COMPUCELL3D includes viscous dissipation explicitly.

E includes terms to describe each biological mechanism that we will employ in a model, e.g.:

$$E = E_{Contact} + E_{Volume} + E_{Chemical}. \tag{5}$$

We describe each of these terms below:

1. **Cell-Cell Adhesion:** Cells have numerous classes of adhesion molecules on their membranes which are responsible for both specific and nonspecific binding to other cells and to the ECM. The great variability of these molecules means that cells can control in detail their binding energy to each class of partner which they are likely to encounter and whether this binding is reversible or irreversible. The binding molecules also generally act as receptors, i.e., binding of a molecule to its ligand can induce a signaling cascade within the cell that can cause the cell to modify its behavior.

   In (5), $E_{Contact}$ describes the net adhesion/repulsion between two cell membranes. It is the product of the *binding energy per unit area*, $J_{\tau,\tau'}$, and the total area. $J_{\tau,\tau'}$ depends on the *types* of the interacting cells, τ and τ'. The equation for $E_{Contact}$ is:

$$E_{Contact} = \sum_{(i,j,k),(i',j',k')neighbors} J_{\tau(\sigma),\tau'(\sigma')} (1 - \delta(\sigma(i,j,k),\sigma'(i',j',k'))), \tag{6}$$

   where the *Kronecker delta* $\delta(\sigma,\sigma') = 0$ if $\sigma \neq \sigma'$ and $\delta(\sigma,\sigma') = 1$ if $\sigma = \sigma'$, ensuring that only links between surface sites in different cells contribute to the cell-adhesion energy.
2. **Cell Growth, Division, and Death:** A cell of type τ has a prescribed *target volume* $v(\sigma,\tau)$ and *target surface area* $s(\sigma,\tau)$. Translating actual volumes to CPM target volumes involves fixing the ratio between the CPM lattice size in pixels and the actual domain length. The actual volume and surface area fluctuate around these target values, e.g., due to changes in osmotic pressure, pseudopodal motion of cells, etc. Changes also result from growth and division of cells during morphogenesis. E_{Volume} enforces these targets by exacting an energy penalty for deviations from the targets. E_{Volume} depends on four model parameters: *volume elasticity*, λ, target volume, $v_{target}(\sigma,\tau)$, *membrane elasticity*, λ', and target surface area, $s_{target}(\sigma,\tau)$:

$$E_{Volume} = \sum_{cells} \lambda_\sigma (v(\sigma,\tau) - v_{target}(\sigma,\tau))^2 + \sum_{cells} \lambda'_\sigma (s(\sigma,\tau) - s_{target}(\sigma,\tau))^2. \tag{7}$$

   We model cell growth by allowing the values of $v_{target}(\sigma,\tau)$ and $s_{target}(\sigma,\tau)$ to increase with time. Cell division occurs when the cell reaches a fixed, type-dependent volume. We model division by starting with a cell of average size, $v_{target} = v_{targetaverage}$, causing it to grow by gradually increasing v_{target} to $2v_{targetaverage}$ and splitting the dividing cell into two cells, each with a new target volume $v_{target}/2$. The axis of division may either be random or oriented perpendicular to the long axis of the cell; in our applications of the model we use the former. One daughter cell assumes a new identity (a unique value of σ). We model cell death simply by setting the cell's target volume and target surface area to zero.
3. **Chemotaxis and Haptotaxis:** Cells can respond to chemical signals by moving along diffusible or substrate-bound concentration gradients of a signal molecule. The first mechanism is *chemotaxis*, the second is *haptotaxis*. A chemotaxis model requires a representation of the evolving and spatially-varying chemical-concentration field and a model mechanism linking the field to the framework for cell and tissue dynamics. The former depends on the particular morphogen molecule. $C(x,y,z)$ is the local concentration of the morphogen molecule in extracellular space. An effective chemical potential $\mu(\sigma)$ models chemotaxis or haptotaxis to incorporate the effective chemical energy into the CPM energy formalism:

$$E_{Chemical} = \sum_{x,y,z} \mu(\sigma(x,y,z))C(x,y,z), \tag{8}$$

   for a linear response. Higher-order responses are also possible as we showed above (see (3)) [62].

   Haptotaxis resembles chemotaxis in (8), but $C(x,y,z)$ does not diffuse. Our current chemotaxis and haptotaxis implementation makes two strong simplifications: that the chemical fields interpenetrate the cells and are tied to the underlying CPM lattice. The RAD-solver release of COMPUCELL3D will allow for excluded volumes and advection due to cell and ECM movement. However, the extensive and successful simulations of the Utrecht group [55], [56], [77], [61], [62], [42], [41], [40], [57], [58] show that neither of these simplifications fundamentally affects behavior when movement is slow compared to diffusion rate (as it is in most developmental biology simulations).

3.2 Reaction-Diffusion

Turing [85] introduced the idea that interactions of reacting and diffusing chemicals (usually of two species denoted u_1 and u_2) could form self-organizing instabilities that provide the basis for biological patterning. We use his continuum, PDE RD approach. For simplicity, we assume isotropic

diffusion (i.e., d_j^i, the components of the diffusion terms, do not depend on i or j), so:

$$\frac{\partial \vec{u}}{\partial t} = D\nabla^2\vec{u} + F(\vec{u}), \tag{9}$$

where $\vec{u} = (u_1, u_2)^T$ and $D = diag(d_1, d_2)$, where d_1 and d_2 are the diffusion constants for u_1 and u_2, respectively. Without loss of generality, we can assume that $d_1 = 1$, and $d_2 = d$. The term $F(\vec{u})$ describes the reaction kinetics.

3.3 Cell Type and State

During morphogenesis, cells *differentiate* from initial multipotent stem cells into the specialized types of the developed organism. Though every cell is different, identifying cells with broadly similar behaviors and grouping them into *differentiation types* is standard practice in biology. Cell differentiation from one cell type to another is a comprehensive, qualitative change in cell behavior, generally abrupt and irreversible (e.g., responding to new sets of signals or turning on or off whole genetic pathways). All cells of a particular differentiation type share a set of parameters describing their state, while two different cell types (e.g., muscle and bone) have different parameter sets. Cells of the same type can also exist in different *states*, corresponding to a specific set of values for the parameter set of the cell type. A cell's behavior depends on its state; if all parameters associated with their cell type were exactly the same, two model cells would behave identically in the same external environment, while cells of the same type with different parameter values would behave differently.

In developmental processes like the ones we consider, networks of autoregulatory transcription factors common to all cells of a particular type form multistable, multistate, dynamical systems based on their ability to switch arrays of cell-type-specific genes on and off [49].

Here, we model differentiation using a *type-change* map, representing a state automaton. Each type in this map corresponds to a cell type (with a defined parameter set) that exists during a particular stage of morphogenesis (see Section 5). Change of a cell from one type to another corresponds to cell differentiation. The type-change map models regulatory networks by defining the rules governing type changes, which take into account the intracellular and intercellular effects of chemical fields (see [28], [74] for formal issues involved in creating cell-type-transition maps from continuum and discrete approximations of complex gene regulatory networks). Other approaches to modeling gene regulatory networks are possible [48].

4 Motivation behind CompuCell3D

We originally developed a 2D engine for morphogenesis called CompuCell [44]. This work extends CompuCell to 3D, increasing the range and biological realism of CompuCell3D simulations. We have improved the engine's efficiency through better data structures and algorithms, and its extensibility through a more thoroughly object-oriented design, which uses scientific design patterns [79], [14].

Specifically, CompuCell consumed too much memory when running 3D simulations. For example, we could not extend the technique of representing grid space as a 2D array to 3D because of the quantity of memory such an array consumed. Consider a relatively small 200^3 grid, with each pixel consuming a very conservative 32 bits. In three dimensions, this grid requires approximately 30 MB of memory, compared to only 156 KB for a 200^2 grid. To reduce memory usage, CompuCell3D implements conservative grid allocation, which only allocates space to a grid pixel if the pixel belongs to a cell and otherwise points to a singleton representing the surrounding medium. This technique eliminates memory allocation for potentially hundreds of thousands of pixels and is one of multiple techniques we use for careful memory management.

Paging causes a second memory issue. The Metropolis algorithm attempts pixel-index flips hundreds of thousands to billions of times per simulation step, requiring new pixel information that many times per step. If the information (for example, attributes) associated with each pixel is heavily scattered in virtual memory, the page-fault rate could skyrocket, with multiple sets of pixels and attributes constantly swapping in and out, greatly degrading performance due to thrashing. We addressed this problem via two techniques: *offset neighbor evaluation* and *contiguous attribute allocation*. The former specifically improves the performance of energy Hamiltonians that need to calculate pixel neighbors (for example, $E_{Contact}$ in the CPM) and of neighbor selection. Offset evaluation finds neighbors for a pixel only when necessary, and caches neighbor pixels in an array for later use by the same or different pixels in the grid. Contiguous attribute allocation takes a grid point, and if it forms part of a cell, stores a pointer to a location in memory which contiguously stores the parameter set representing the state of that cell. This technique reduces the page-fault rate by storing related information within a single page, reducing the number of pages swapped in when we reference a cell in the grid.

Certain programming languages (e.g., Fortran [86]) have built-in features for contiguous allocation that could benefit CompuCell3D. However, Fortran lacks flexibility. An object-oriented language provides a solid basis for a flexible framework through *polymorphism* (allowing objects that share common logic to inherit methods and data members from a predefined interface), so we implemented the back end of CompuCell3D in C++ with careful memory management and other techniques to improve flexibility.

Building on polymorphism to achieve flexibility, we can add new functionality to CompuCell3D using six different simulation objects, each with its own predefined interface:

1. **Energy function**: Computes energies used by the CPM. An example is $E_{Contact}$ ((6)), implemented as a `ContactEnergy` energy function in the CompuCell3D source.
2. **Acceptance function**: Computes the probability of accepting a CPM pixel flip. An example is the Metropolis acceptance function of (4).
3. **Steppable**: Provides functionality to execute a routine after every n simulation *Monte Carlo steps* (*MCS*) (a Monte Carlo step attempts N pixel flips, where N is the number of lattice points). An example would be a dumper that outputs grid data for

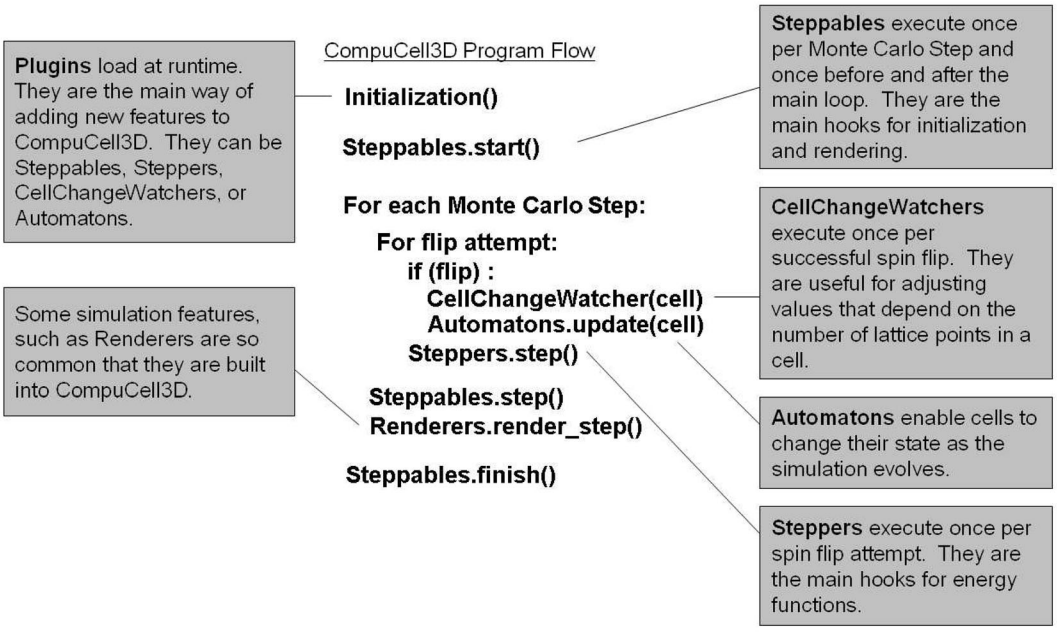

Fig. 3. Pseudocode for the execution of simulation objects. *Diagram by Chris Mueller, Indiana University Bloomington.*

visualization, perhaps, representing chemical concentrations or current cell types.

4. **Cell-Change Watcher**: Provides functionality to execute a routine after every successful pixel flip. A volume calculator is an example, since a pixel flip changes the volume of two cells.
5. **Stepper**: Provides functionality to execute a routine after every pixel-flip attempt. A sample application is a deallocator of memory for dead cells (those with zero volume). We perform the zero-volume check in a cell-change watcher and set a flag. We then check the flag in a stepper and deallocate if necessary. This sequence avoids null references if the flip has additional watchers to execute.
6. **Plugin**: Encapsulates the functionality of a combination of the previous objects. An example of this situation is the implementation of cell mitosis. A cell divides if its volume exceeds the global doubling volume. Therefore, a successful pixel flip (cell-change watcher) requires checking the volume of the cell with the added pixel and setting a Boolean flag if the cell's volume is higher than the doubling volume. The stepper checks the flag before the next pixel-flip attempt and invokes the cell mitosis code if the flag is set. This sample plugin would inherit from two interfaces: the cell-change watcher and the stepper, as a result possessing both their abilities.

Fig. 3 describes the execution order of these simulation objects, which work in conjunction with a COMPUCELL3D XML input configuration file that provides input elements, normally by the addition of a few lines of code. Plugins and steppables implement functions to read from the XML configuration file and so can accept input variables and values from it. The benefit of a configuration file extensible by a single method comes with the inherent cross-platform compatibility of XML, increasing flexibility by permitting the use of COMPUCELL3D on machines running different operating systems.

While this structure for adding new simulation objects coded in C++ and interfacing them to the framework through the configuration file lays a solid groundwork for simple extensibility, we realize that many biocomplexity researchers and potential users of COMPUCELL3D may not be experienced C++ programmers. For this reason, we have interfaced COMPUCELL3D with BIOLOGO [19], an XML-based domain-specific language for morphogenesis. We designed the syntax of this language to be easy to understand by researchers studying morphogenesis who wish to use the CPM. BIOLOGO allows users to extend COMPUCELL3D with customizable energy functions, chemical fields, and Cell-Type-Maps. BIOLOGO automatically generates plugins implemented as C++ code.

5 Implementation of Techniques and Biological Concepts in CompuCell3D

This section explains how we translate the various biological phenomena that compose our simulations into a COMPUCELL3D software implementation. We also include a more detailed description of the software techniques that we outlined in the previous section.

5.1 Biological Cell

A cell *factory* creates COMPUCELL3D biological cells. Factories are useful techniques in highly polymorphic, object-oriented design due to their runtime decisions on derived-class object creation and deletion [6]. We use the BasicUtils library, which contains an implementation of a `BaseDynamicClassFactory` with `virtual` functions for allocation and deallocation analogous to `new` and `delete` functions in C++ [20]. This organization improves software flexibility, since we can now allocate and deallocate an

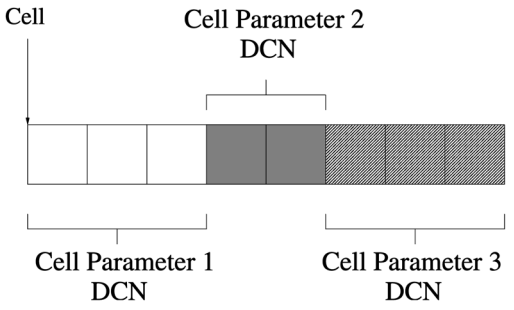

Fig. 4. Representation in memory of a `Cell` object with three attributes represented as dynamic class nodes of size 12, 8, and 12 bytes. Each cell takes 32 bytes of contiguously allocated memory. This figure assumes one-word blocks.

object of a derived class without specifying the actual derived class to which it belongs.

A pointer to a set of contiguous memory locations that encompasses all elements of a cell's parameter set represents the basic `Cell` unit. Parameters include center-of-mass coordinates, volume, surface area, etc. A *dynamic class node* (*DCN*) represents each parameter. A DCN improves flexibility by allowing users to define customized `init()` methods that execute parameter-value initialization for each individual DCN, and aids memory management by allowing access to a given DCN via a specific pointer and offset.

The user can add parameters to each cell by *registering* new DCNs. A registered DCN stores its size (in bytes) and offset from each `Cell` pointer. We define the offset as the size (in bytes) of the total number of DCNs registered thus far. Fig. 4 shows a schematic of the contiguous memory allocation for a cell, assuming one-word blocks.

COMPUCELL3D performs this allocation for each `Cell` in the simulation. We can access `Cell` parameters represented as DCNs by supplying the `Cell` pointer and the name of the DCN, since the DCN stores the offset from the `Cell` pointer.

In this scheme, if the amount of memory a `Cell`'s attributes consume is small enough to fit into one virtual page, the fragmentation within each `Cell` object is zero. The method also reduces global external fragmentation. Consider a simulation with a large number of `Cells`, each with large parameter sets. If the memory allocation for each individual parameter set for each `Cell` is not contiguous, thousands of parameters of varying sizes will be scattered in memory, creating many small holes and potentially requiring frequent compaction. On the other hand, using DCNs for parameters enforces contiguous parameter set allocation for each `Cell`, as long as all attributes can fit into one page. Thus, data sets for `Cells` lie scattered in memory, rather than individual parameters for each `Cell`, a much better granularity. Contiguous allocation also allows us to take advantage of spatial locality which can drastically reduce page faults.

5.2 Cell Grid and Watchers

We implement the 3D cell grid as a resizable array of `Cell` object pointers called a *field*, defined by the class `Field3D`. We used conservative grid allocation. After CPM pixel flips, pointers to the medium singleton may change to point to `Cell` objects, and *vice versa*.

Because some operations must execute after every change in the `Cell` field, we must keep track of, or *watch*, the `Cell` field. Therefore, we declare the `Cell` field not just as a `Field3D` object, but as a `WatchableField3D` object—the only difference being that a `WatchableField3D` object keeps an array listing its "*watchers*." We represent watchers as `CellChangeWatcher` objects. Each watcher defines a method `field3DChange()` which defines the appropriate actions to take after a change in the `Cell` field. After a `Cell`-field update, we pass through the array of watchers and invoke their `field3DChange()` methods. Watcher examples include cell-volume, surface-area, or center-of-mass calculators.

A simulation can contain many `Cell` objects. Consider the impact of conservative grid allocation on the previous example with 32 byte `Cells`. In this case, allocating memory only as needed in the `Cell` grid and having each medium grid point reference a singleton, saves 536,722 `Cell`-object allocations. The total memory all `Cells` consume in the simulation is roughly 81 KB versus up to 16 MB for naive allocation, a savings of about 95 percent. The savings becomes even larger for larger grids.

5.3 Energy Computation and Arbitrary Neighbors

The CPM implements a physical description of cells based on the principle of energy minimization. A CPM simulation must account for many different energies (e.g., contact, volume/surface, chemical). Energy functions within COMPUCELL3D inherit from an abstract class `EnergyFunction`, which contains a `virtual` function `changeEnergy()`, which the user defines for each energy function. A single call to a method `registerEnergyFunction()` *registers* an `EnergyFunction`, passing the `EnergyFunction` object as a parameter. Invocation of this method tells COMPUCELL3D to include this energy calculation when deciding whether or not to flip the index of a selected pixel.

Some energy functions (e.g., contact energy) require a grid pixel to compute its interaction with neighboring pixels. To find pixels neighboring a grid point, we implement a `NeighborFinder` singleton which uses offset evaluation. Algorithm 1 gives pseudocode for the offset neighbor evaluation. The algorithm for offset evaluation assumes that a pixel's neighbors lie within some small constant distance D. Knowing this distance, we first find neighbors *to the origin*, by finding points (X, Y, Z) such that $\sqrt{X^2 + Y^2 + Z^2} \leq D$. We find other neighbors at this distance by rotating point (X, Y, Z) about the origin. Then, for a specific point (x, y, z) we translate these neighbors by adding (x, y, z) to their coordinates, giving us the neighbors of (x, y, z).

Algorithm 1 Pseudocode for offset neighor evaluation

Neighbor Finder:

1) Preprocessing: Initialize $x := 0$ and $neighbor\_array$ to be empty.
   $neighbor\_array$ is an array of pairs of points and integer distances, and $n := 0$;
2) **getNeighbor(int** n, **double &**D)

a) **while** length of $neighbor\_array < n$
  i) $x := x + 1$;
  ii) **for** each (X, Y, Z) such that $x = X^2 + Y^2 + Z^2$
    A) **for** each unique point Q that is a rotation of (X, Y, Z) around the axes
      Add $(Q, \sqrt{x})$ to $neighbor\_array$;
b) $D := neighbor\_array[n]$.distance;
c) **return** $neighbor\_array[n]$.point;

3) To look at level 1 neighbors, distance 1 from a point P:
a) **do**
  i) $neighbor$ = getNeighbor(n, D) + P;
  ii) ... Do something with neighbor ...
  iii) $n := n + 1$;
  while $D <= 1$

Finding the first n neighbors of a given point (x, y, z) requires D^2 integer iterations, where D is the distance within which all n neighbors lie. For each $j \leq D$, we test all possible integer values of X, Y, Z between 0 and j. If $X^2 + Y^2 + Z^2 = j^2$, we add the neighbors at distance j to a neighbor array, until we reach n neighbors. In this way, we insert the neighbors into the neighbor array in order of distance. Because we calculate neighbors with respect to the origin, we can reuse them to find the neighbors of multiple grid pixels. Since executing this entire algorithm to calculate each neighbor is prohibitively slow, we cache the neighbors into an array for later use. With offset evaluation and dynamic array growth, we calculate neighbors with value n only as needed.

This lazy evaluation technique for calculating arbitrary neighbors of a pixel greatly improves simulation speed. To illustrate this improvement, compare a 3D CPM algorithm with offset evaluation (program B) to a different version (program A) that forces each grid pixel to maintain pointers to all first, second, third, and fourth neighbors. We ran the two versions on a PC with an AMD Athlon XP 1800+ at 1.6 GHZ, and 512 MB of memory running RedHat Linux 9.0, kernel 2.4.22. The field dimensions were $71 \times 36 \times 211$ pixels with a cell size of $2 \times 2 \times 2$, an initially uniform cell distribution, temperature of 1.0, data output every 10 steps, and 539,316 flip attempts per simulation step. We used contact, volume, and chemical energies for energy computation in the CPM algorithm, and turned off visualization to restrict performance measurement to computation. We measured times using "real" or wall-clock times using the Linux `time` command.

Version A has a much longer startup time than program B. Subtracting this initial startup time (time for the first timestep), the timing and memory usage are:

	A	B	Ratio A/B
Execution Time For 100 CPM Flips	1959 s	501 s	3.91
Memory Usage	70656 KB	6564 KB	10.76

Therefore, even without the expensive initialization costs in version A, the offset neighbor evaluator yields a four-fold speedup in computation. The offset evaluator for neighbors also yields a 10-fold memory savings, since a pixel in memory does not need pointers to all neighbors.

The current implementation of COMPUCELL3D includes just one acceptance function, which implements the Metropolis algorithm with a Boltzmann acceptance function, but we can create custom acceptance functions, for example to implement Kawasaki or Glauber dynamics [8], [92].

5.4 Cell-Type-Map

COMPUCELL3D includes a cellular automaton to implement a Cell-Type-Map. Each individual type of cell has three methods defined:

1. A method for initializing state variables,
2. a method for updating states (changing state-variable values), and
3. a method for changing type.

Cells of different types may react differently to external and internal conditions. Therefore, the definitions of each of these methods will vary with cell type. COMPUCELL3D changes cell types after each CPM step, so we implement the automaton as a `TypePlugin`. When the CPM selects a pixel candidate, it finds the corresponding cell and invokes the appropriate methods for updating its state and type. If the cell's type changes, we reinitialize the state variables appropriately for the new type.

5.5 External Chemical Concentration

COMPUCELL3D implements both resizable field structures and platform-independent file reading of external chemical concentrations. Various simulation objects such as plugins, steppables, or steppers can update these fields.

5.6 Boundary Conditions

COMPUCELL3D implements *NoFlux* and *Periodic* boundary conditions on each individual axis. Calculating neighboring pixels using NoFlux boundary conditions involves discarding pixels outside the lattice. If a neighbor pixel lies outside the lattice using Periodic boundary conditions, modular arithmetic implements wraparound.

5.7 Flexibility

We can extend COMPUCELL3D by encapsulating new functionality (e.g., new energy functions, new cell-change watchers, or new rules for cell differentiation and state) into one of the six simulation objects: plugins, steppables, steppers, acceptance functions, energy functions, and cell-change watchers. COMPUCELL3D accepts a configuration file as input. This configuration file can add or remove plugins and steppables, as well as 3D graphical-field renderer objects, from simulations. The COMPUCELL Web page [21] provides complete COMPUCELL3D configuration-file syntax along with configuration files for this paper's verification simulation and two others: basic cell sorting [80], [81], [11], [32] and an avian limb without domain growth [71], [44], [16].

COMPUCELL3D can also read a *Potts Initial File* (*PIF*) for initial cell positions. The format of the PIF is straightforward, with repeated lines of the form:

$$< \texttt{cell\#} > < \texttt{celltype} > \texttt{x1 x2 y1 y2 z1 z2},$$

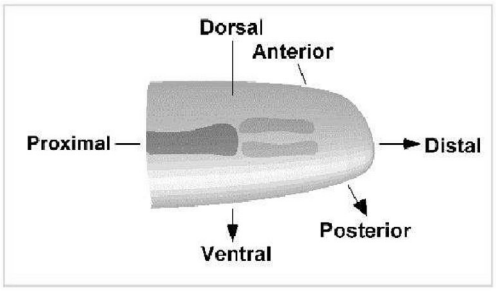

Fig. 5. Schematic diagram of chick limb organogenesis.

resulting in a rectangular cell extending from ($x1$, $y1$, $z1$) to ($x2$, $y2$, $z2$). Reuse of the same cell # allows arbitrary cell shapes.

6 Verification Simulation

Chaturvedi et al. [18] and Izaguirre et al. [44] used the systems-biology approach of integrating discrete and continuum models for biological mechanisms to describe a reduced, 2D model of vertebrate limb development. Fig. 5 schematically represents the major axes and the progress of chondrogenesis in a developing vertebrate forelimb at a stage part-way through development. The humerus (in dark gray) has already differentiated into cartilage, the radius and ulna (light gray) are beginning to form, and the wrist bones and digits are still to form.

Here, we use discrete models to describe cell movement, division, interactions, and differentiation from multipotent cells into specific cell types. We use the modified form of the equations of Alber et al., described above (1), extending this originally 2D RD model to 3D [38]. The concentration of the activator chemical, in response to which in our model (although not in the model of Hentschel et al.) cells undergo chemotaxis and increase their adhesivity levels, occupies a second matching grid. We measure activator concentration for a cell grid pixel using its corresponding location in the concentration grid. High activator concentration induces production of a second secreted molecule, which we identify with fibronectin. As soon as the TGF-β concentration at a grid pixel exceeds a threshold, the corresponding cell secretes fibronectin at that pixel location, at a user-specified rate. Fibronectin concentration, in turn, supplies the effective chemical energy for haptotaxis in (8).

We superimpose a third chemical, FGF, on the grid to control growth and RD. FGF concentration monotonically increases with z across the entire grid, with normalized values between 0.1 and 1. A domain known as the *Apical Zone*, within which no RD can occur, moves proximo-distally within the grid. Fig. 6a shows a sketch of a 2D example of one step of the growth algorithm starting from a uniform cell distribution.

Three-dimensional simulations add biological realism. In 2D, for example, the Apical Zone is crucial to the simulation and allows cells to move perpendicularly to the proximo-distal direction. In 2D without the Apical Zone, the noncondensing zone becomes disconnected, and one part of the limb always grows faster than the rest, see Fig. 7. In our 3D simulation, the noncondensing zone remains connected, with or without the Apical Zone.

Initially, we distribute cells uniformly in a user-specified fraction of the overall grid (with respect to z) and the Apical Zone occupies this grid fraction. When the grid grows by n rows in the positive z direction, the Apical Zone shifts upward in the positive z direction by n rows, allowing cells in the *Active Zone* below it to react and form patterns. We specify a variable `FGFThreshold` which sets the size of the Apical Zone. The z value in the grid where the FGF concentration equals this threshold defines the proximal boundary of the Apical Zone, and the z value at which the FGF concentration is maximal defines the *Apical Ectodermal Ridge* (*AER*), the distal boundary of the Apical Zone. When a cell attempts to react to the surrounding activator we check the FGF concentration at its location in the grid. The cell cannot react if the concentration is greater than or equal to `FGFThreshold` (implying that it is in the Apical Zone). The grid thus divides into two zones, the Apical Zone and the Active Zone. The FGF concentration changes linearly *within each zone* (see Fig. 6b):

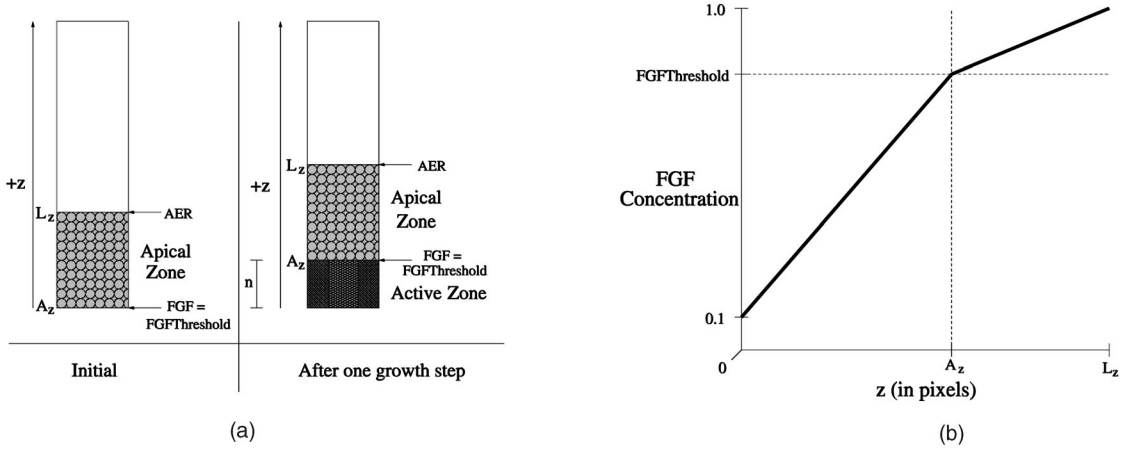

Fig. 6. (a) One step of the growth algorithm applied to the chick-limb simulation. Cells are initially uniformly distributed and occupy a third of the grid, as does the Apical Zone. After one growth step, the Apical Zone shifts up by n rows in the z direction, allowing cells in the newly formed Active Zone to condense into patterns. (b) The distribution of FGF with respect to the z-coordinate of the grid point.

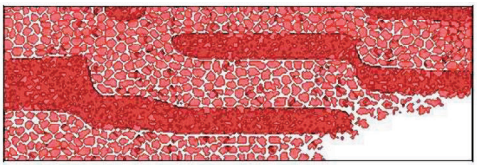

Fig. 7. 2D simulation without the *Apical Zone*: one side grows faster than the other because the noncondensing zone is not connected.

$$FGF(z) = \begin{cases} \frac{z-A_z}{L_z-A_z}(1-F_t)+F_t, & z \geq A_z, \\ \frac{z}{A_z}(F_t-0.1)+0.1, & z < A_z. \end{cases} \quad (10)$$

A_z is the z-coordinate of the lower boundary of the Apical Zone, L_z the z-dimension of the grid itself, F_t the `FGFThreshold`, and z the proximo-distal position.

TGF-β also plays a role in our customized Cell-Type-Map. Cells can be of type `NonCondensing` or `Condensing`, with `Condensing` cells being more adhesive. When a cell outside the Apical Zone occupies a point in the grid where the TGF-β concentration exceeds a threshold, the cell becomes `Condensing`, otherwise the cell is `NonCondensing`. Cells are initially `NonCondensing` and cells cannot condense within the Apical Zone. Fig. 8 shows the state diagram for these cell types. Only `Condensing` cells have an energy term for haptotaxis to fibronectin (8).

We implement the algorithm for domain growth in COMPUCELL3D as a `Steppable` object. We define the cell *Density* as follows:

$$Density = \frac{C}{T} \times 100, \quad (11)$$

where C is the total number of pixels which contain cells, T is the total number of pixels, and we specify a *range box*, the domain over which to calculate the density. We also specify the input variables `delay` (in timesteps), `threshold` (percentage), and `n` (number of rows to grow at a time) for simulations with growing domains. If the density within the range box exceeds the `threshold`, the mathematical grid grows by `n` rows in the z direction. A growth step turns off cell mitosis for `delay` steps, and the algorithm repeats. This `delay` allows the cells time to cluster and fill in the `n` initially empty, added rows. Fig. 9 shows the domain-growth algorithm in COMPUCELL3D.

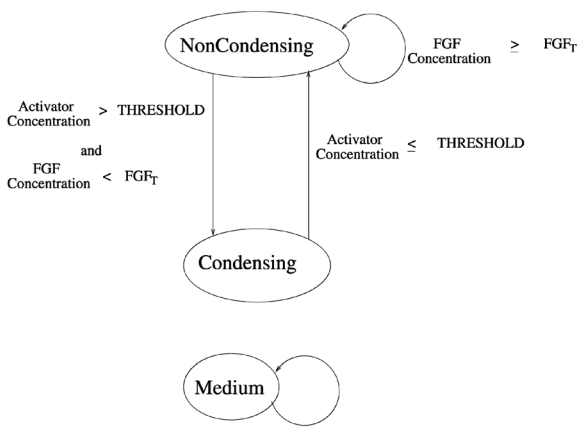

Fig. 8. Cell-Type-Map implemented using COMPUCELL3D for the sample simulation.

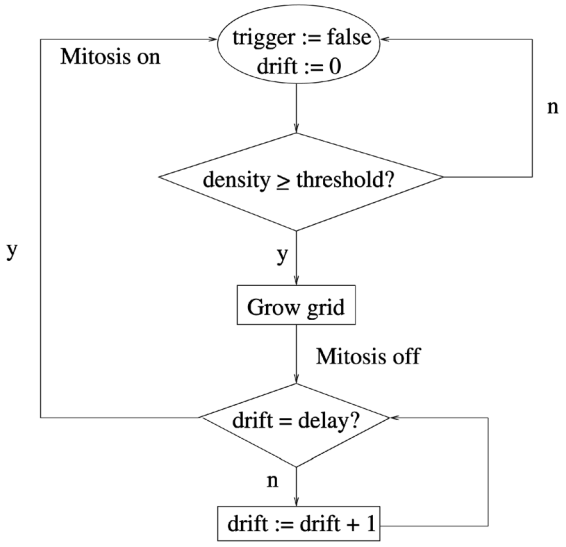

Fig. 9. Growth algorithm.

Mitosis rates come from [54]. The volume of a mesenchymal cell of approximately 15 μm diameter is 1.8×10^3 μm^3 and its surface area is approximately 1.8×10^2 μm^2. The simulation starts with eoght cells, and eventually grows to contain 13,902 cells. Since the typical CPM cell has a target volume of 120 pixels, the length scale is 3 μm per lattice site.

Fig. 10a shows our simulation of 3D limb growth and pattern formation at 2250 MCS, 3,250 MCS, and 4,250 MCS, visualized using Ogle [72]. We ran the simulation for 4,250 MCS, which took 7 hours, 1 minute, and 13 seconds on a 2.4 GHz Intel Xeon processor with 1 GB of RAM. We show three screenshots and for clarity have superimposed the `Condensing` cells (gray) on the grid containing all the cells. `NonCondensing` cells are red. Note the Apical Zone (pink cells) at the distal end of the limb, where no cell condensation can occur. Fig. 10b shows elliptical cross-sections of the fully formed limb and illustrates the formation of cylindrical chondrogenic structures. This model represents an extension and improvement of that of Hentschel et al. [38], in that it represents the entire limb in three dimensions and its constituent cells as extended objects which react to their complex microenvironments. Like the earlier model, however, it generates a simplified, but clearly limb-like, skeletal pattern solely on the basis of experimentally-confirmed cell-biological ingredients and interactions and well-established biophysical processes. Simulations with different parameter choices from those we show here confirm that the model would reproduce experimental manipulations such as implantation of a source of the morphogen *Sonic hedgehog*, which widens the limb tip and, consequently, leads to extra digits [75]. In particular, in related studies we show that changes in the parameters corresponding to known genetic or surgical alterations in the core components of the simulation lead to biologically-observed outcomes, such as fusion of skeletal elements (see Fig. 2 in [16]).

Fig. 10. (a) 3D chicken limb growth and patterning visualized with Ogle. On top of a visualization of all cells, we superimpose one showing only `Condensing` cells (in gray) for clarity. Apical Zone cells are pink. No condensation occurs in the Apical Zone. (b) Thick cross sections of fully grown avian limb bud from (a). Once again, we superimposed the `Condensing` cells for clarity. Color scheme is the same as in (a); `NonCondensing` cells are red and `Condensing` cells are gray.

7 Conclusions

We have presented an extended, 3D version of the CompuCell modeling environment. CompuCell3D is faster and uses much less memory than CompuCell. CompuCell3D computes the time evolution of differentiating cells in 3D using a stochastic CPM and a Cell-Type-Map implemented as a discrete cellular automaton. Continuum reaction-diffusion equations model chemical signaling.

We have verified the correctness of CompuCell3D by simulating skeletal patterning in the avian limb. Our RD equations model a core cellular-biochemical network involving activating (TGF-β) and inhibitory morphogens, the ECM protein fibronectin, a growth factor (FGF) and its receptors, all of which experiments show to be essential to limb development. From this model, using realistic spatial scales for our lattice, we observe the emerging spatiotemporal development of the skeleton. The Cell-Type-Map uses biological information about chondrogenesis to model the transition from mesenchymal to condensing cells. Finally, the CPM uses relative cell-adhesion parameters that surface-tension experiments on different cell types can determine. The literature provides volumes and mitosis rates. In related work we have shown how our model can predict skeletal element fusion [16].

We can extend CompuCell3D directly using C++ plugins, or more abstractly by describing arbitrary Cell-Type-Maps, chemical fields, and energy functions using BIOLOGO, an XML-based domain-specific language for morphogenesis.

We are currently extending CompuCell3D to integrate more realistic geometry, specifically irregular grids and moving boundaries. These involve more careful pixel and neighbor-selection algorithms, which must discard pixels outside the moving boundary. We have also developed a RAD solver which runs with the CPM [23].

We are creating a parallel version of CompuCell3D, which will more easily accommodate very large and long simulations. Parallelization is somewhat challenging because an event-driven, kinetic Monte Carlo process underlies CompuCell3D, requiring complex block-parallelization algorithms [91], [70], [82].

Acknowledgments

This research was partiall funded by US National Science Foundation grants IBN-0083653, IBN-0313730, and ACI-0135195. James A. Glazier acknowledges support from NASA grant NAG3-1619, an IBM Innovation Institute Award, and a Pervasive Technology Laboratories Fellowship. Trevor Cickovski was funded by an Arthur J. Schmitt Fellowship. Special thanks to Kedar Aras, Joseph Coffland, Chris Mueller, and Todd Schneider for their contributions to the design of CompuCell3D.

References

[1] M.S. Alber, T. Glimm, H.G. E. Hentschel, B. Kazmiercazk, and S.A. Newman, "Stability of N-Dimensional Patterns in a Generalized Turing System: Implications for Biological Pattern Formation," *Nonlinearity*, vol. 18, no. 1, pp. 125-138, 2005.

[2] M.S. Alber, H.G.E. Hentschel, B. Kazmiercazk, and S.A. Newman, "Existence of Solutions to a New Model of Biological Pattern Formation," *J. Math. Analysis and Applications*, 2005.

[3] M.S. Alber, Y. Jiang, and M.A. Kiskowski, "Lattice Gas Cellular Automaton Model for Rippling and Aggregation in Myxobacteria," *Physica D*, vol. 191, nos. 3-4, pp. 343-358, 2004.

[4] M.S. Alber, M.A. Kiskowski, J.A. Glazier, and Y. Jiang, "On Cellular Automaton Approaches to Modeling Biological Cells," *IMA Math. Systems Theory in Biology, Comm., and Finance,* J. Rosenthal and D.S. Gilliam, eds., vol. 134, pp. 1-39, Springer-Verlag, 2003.

[5] M.S. Alber, M.A. Kiskowski, and Y. Jiang, "Two-Stage Aggregate Formation via Streams in Myxobacteria," *Physical Rev. Letters,* vol. 93:068301, 2004.

[6] A. Alexandrescu, *Modern C++ Design: Generic Programming and Design Patterns Applied.* Reading, Mass.: Addison-Wesley, 2001.

[7] A. Arkin, J. Ross, and H.H. McAdams, "Stochastic Kinetic Analysis of Developmental Pathway Bifurcation in Phage Lambda-Infected Escherichia Coli Cells," *Genetics,* vol. 149, no. 4, pp. 1633-1648, 1998.

[8] S. Artz and S. Trimper, "Competing Glauber and Kawasaki Dynamics," *Int'l J. Modern Physics B,* vol. 12, no. 23, pp. 2385-2392, 1998.

[9] M. Asipauskas, M. Aubouy, J.A. Glazier, F. Graner, and Y. Jiang, "A Texture Tensor to Quantify Deformations: The Example of Two-Dimensional Flowing Foams," *Granular Matter,* vol. 5, no. 2, pp. 71-74, 2003.

[10] J.B. Bassingthwaighte, "Strategies for the PHYSIOME Project," *Annals of Biomedical Eng.,* vol. 28, no. 8, pp. 1043-1058, 2000.

[11] D.A. Beysens, G. Forgacs, and J.A. Glazier, "Cell Sorting is Analogous to Phase Ordering in Fluids," *Proc. Nat'l Academy of Science,* vol. 97, no. 17, pp. 9467-9471, 2000.

[12] D.A. Beysens, G. Forgacs, and J.A. Glazier, "Embryonic Tissues are Viscoelastic Materials," *Canadian J. Physics,* vol. 78, no. 3, pp. 243-251, 2000.

[13] BioSPICE, Biospice Community Web Site: Biology in Silico, https://community.biospice.org, 2005.

[14] C. Blilie, "Patterns in Scientific Software; An Introduction," *Computing in Science and Eng.,* vol. 4, no. 3, pp. 48-53, 2002.

[15] Cell-O-Sim, Cell-O-Sim Web site, http://mbi.dkfz-heidelberg.de/projects/cellsim/cellosim, 2005.

[16] R. Chaturvedi, C. Huang, J.A. Izaguirre, S.A. Newman, J.A. Glazier, and M.S. Alber, "A Hybrid Discrete-Continuum Model for 3-D Skeletogenesis of the Vertebrate Limb," *Lecture Notes in Computer Science,* vol. 3305, pp. 543-552, Springer Verlag, 2004.

[17] R. Chaturvedi, C. Huang, B. Kazmierczak, T. Schneider, J.A. Izaguirre, S.A. Newman, J.A. Glazier, and M.S. Alber, "On Multiscale Approaches to 3-Dimensional Modeling Of Morphogenesis," *J. Royal Soc. Interface,* http://www.nd.edu/malber/Interface_Alber.doc, 2005.

[18] R. Chaturvedi, J.A. Izaguirre, C. Huang, T. Cickovski, P. Virtue, G. Thomas, G. Forgacs, M.S. Alber, H.G.E. Hentschel, S.A. Newman, and J.A. Glazier, "Multi-Model Simulations of Chicken Limb Morphogenesis," *Lecture Notes in Computational Science,* vol. 2659, pp. 39-49, Springer-Verlag, 2003.

[19] T. Cickovski and J.A. Izaguirre, "Biologo, A Domain-Specific Language for Morphogenesis," *ACM Trans. Programming Languages and Systems,* http://www.nd.edu/tcickovs/acmtr2e.pdf, 2005.

[20] J. Coffland,2003, Basicutils, http://compucell.sourceforge.net/phpwiki/index.php/BasicUtils.

[21] COMPUCELL, "COMPUCELL: A Framework for Three-Dimensional Simulation of Morphogenesis,"http://sourceforge.net/projects/compucell/, Sept. 2004.

[22] E.J. Crampin, W.W. Hackborn, and P.K. Maini, "Pattern Formation in Reaction-Diffusion Models with Nonuniform Domain Growth," *Bull. Math. Biology,* vol. 64, no. 4, pp. 747-769, 2002.

[23] D. Dan and J.A. Glazier, "Study of Diffusion and Chemotaxis in Cell Motion Using Cellular Potts Model," *Physical Rev. E,* 2005.

[24] DARPA, Darpa news release, *DARPA Releases BioSPICE Software,* https://community.biospice.org/public/bio_release.pdf, 2002.

[25] R. Dillon and H.G. Othmer, "A Mathematical Model for Outgrowth and Spatial Patterning of the Vertebrate Limb Bud," *J. Theoretical Biology,* vol. 197, no. 3, pp. 297-330, 1999.

[26] D. Duguay, R.A. Foty, and M.S. Steinberg, "Cadherin-Mediated Cell Adhesion and Tissue Segregation: Qualitative and Quantitative Determinants," *Developmental Biology,* vol. 253, no. 2, pp. 309-323, 2003.

[27] E-Cell, Inst. for Advanced Biosciences: E-cell Project, http://www.e-cell.org, 2005.

[28] R. Edwards, H.T. Siegelmann, K. Aziza, and L. Glass, "Symbolic Dynamics and Computation in Model Gene Networks," *Chaos,* vol. 11, no. 1, pp. 160-169, 2001.

[29] B. Ermentrout, "Stripes or Spots? Nonlinear Effects in Bifurcation of Reaction-Diffusion Equations on the Square," *Proc. Royal Soc. London A.,* vol. 434, pp. 413-417, 1991.

[30] R.A. Foty, C.M. Pfleger, G. Forgacs, and M.S. Steinberg, "Surface Tensions of Embryonic Tissues Predict Their Mutual Envelopment Behavior," *Development,* vol. 122, no. 5, pp. 1611-1620, 1996.

[31] D.T. Gillespie, "A General Method for Numerically Simulating the Stochastic Time Evolution of Coupled Chemical Reactions," *J. Chemical Physics,* vol. 22, no. 4, pp. 403-434, 1976.

[32] J.A. Glazier and F. Graner, "Simulation of the Differential Adhesion Driven Rearrangement of Biological Cells," *Physical Rev. E,* vol. 47, no. 3, pp. 2128-2154, 1993.

[33] D. Godt and U. Tepass, "*Drosophila* Oocyte Localization is Mediated by Differential Cadherin-Based Adhesion," *Nature,* vol. 395, no. 6700, pp. 387-391, 1998.

[34] A. Gonzalez-Reyes and D. St. Johnston, "The *Drosophila* Ap Axis is Polarised by the Cadherin-Mediated Positioning of the Oocyte," *Development,* vol. 125, no. 18, pp. 3635-3644, 1998.

[35] F. Graner and J.A. Glazier, "Simulation of Biological Cell Sorting Using a Two-Dimensional Extended Potts Model," *Physical Rev. Letters,* vol. 69, no. 13, pp. 2013-2016, 1992.

[36] T. Hayashi and R. Carthew, "Surface Mechanics Mediate Pattern Formation in the Developing Retina," *Nature,* vol. 431, no. 7009, pp. 647-652, 2004.

[37] L.I. Held, *Imaginal Discs: The Genetic and Cellular Logic of Pattern Formation.* New York: Cambridge Univ. Press, 2002.

[38] H.G.E. Hentschel, T. Glimm, J.A. Glazier, and S.A. Newman, "Dynamical Mechanisms for Skeletal Pattern Formation in the Vertebrate Limb," *Proc. Royal Soc. London B Biol. Sci.,* vol. 271, no. 1549, pp. 1713-1722, 2004.

[39] M.L. Hines and N.T. Carnevale, "The NEURON Simulation Environment," *Neural Computation,* vol. 9, no. 6, pp. 1179-1209, 1997.

[40] P. Hogeweg, "Evolving Mechanisms of Morphogenesis: On the Interplay between Differential Adhesion and Cell Differentiation," *J. Theoretical Biology,* vol. 203, no. 4, pp. 317-333, 2000.

[41] P. Hogeweg, "Shapes in the Shadow: Evolutionary Dynamics of Morphogenesis," *Artificial Life,* vol. 6, no. 1, pp. 85-101, 2000.

[42] P. Hogeweg, "Computing an Organism: On the Interface between Informatic and Dynamic Processes," *Biosystems,* vol. 64, nos. 1-3, pp. 97-109, 2002.

[43] V. Hösel and V. Liebscher, "Some Thoughts on the Modeling of Biofilms," *Math. and Statistics,* http://ibb.gsf.de/preprints/1999/pp99-27.ps, 1999.

[44] J.A. Izaguirre, R. Chaturvedi, C. Huang, T. Cickovski, J. Coffland, G. Thomas, G. Forgacs, M.S. Alber, G. Hentschel, S.A. Newman, and J.A. Glazier, "CompuCell, a Multimodel Framework for Simulation of Morphogenesis," *Bioinformatics,* vol. 20, no. 7, pp. 1129-1137, 2004.

[45] Y. Jiang, M. Asipauskas, J.A. Glazier, M. Aubouy, and F. Graner, "*Ab Initio* Derivation of Stress and Strain in Fluid Foams," *Foams, Emulsions and their Applications,* P. Zitha, J. Banhart, and G. Verbist, eds., pp. 297-304, Verlag MIT Publishing, 2000.

[46] Y. Jiang, H. Levine, and J.A. Glazier, "Possible Cooperation of Differential Adhesion and Chemotaxis in Mound Formation of *Dictyostelium*," *European Biophysics J.,* vol. 75, no. 6, pp. 2615-2625, 1998.

[47] Y. Jiang, P. Swart, A. Saxena, M. Asipauskas, and J.A. Glazier, "Hysteresis and Avalanches in Two-Dimensional Foam Rheology Simulations," *Physical Rev. E,* vol. 59, no. 5, pp. 5819-5832, 1999.

[48] H. De Jong, "Modeling and Simulation of Genetic Regulatory Systems: A Literature Review," *J. Computational Biology,* vol. 9, no. 1, pp. 69-105, 2002.

[49] W. Keller, P. König, and T.J. Richmond, "Crystal Structure of a Bzip/DNA Complex at 2.2 Å: Determinants of DNA Specific Recognition," *J. Molecular Biology,* vol. 254, no. 4, pp. 657-667, 1995.

[50] M.A. Kiskowski, M.S. Alber, G.L. Thomas, J.A. Glazier, N.B. Bronstein, J. Pu, and S.A. Newman, "Interplay between Activator-Inhibitor Coupling and Cell-Matrix Adhesion in a Cellular Automaton Model for Chondrogenic Patterning," *Developmental Biology,* vol. 271, no. 2, pp. 372-387, 2004.

[51] M.A. Kiskowski, Y. Jiang, and M.S. Alber, "Role of Streams in Myxobacteria Aggregate Formation," *Physical Biology,* vol. 1, no. 3, pp. 173-183, 2004.

[52] S.P. Kumar and J.C. Feidler, "Biospice, 2," *Omics: A J. Integrative Biology,* vol. 7, no. 4, p. 335, 2003.

[53] K.A. Landman, G.J. Pettet, and D.F. Newgreen, "Mathematical Models of Cell Colonization of Uniformly Growing Domains," *Bull. Math. Biology,* vol. 65, no. 2, pp. 235-262, 2003.

[54] J.H. Lewis, "Fate Maps and the Pattern of Cell Division: A Calculation for the Chick Wing-Bud," *J. Embryol. exp. Morph.,* vol. 33, no. 2, pp. 419-434, 1975.

[55] A.F. M. Marée, "From Pattern Formation to Morphogenesis," PhD thesis, Utrecht Univ., Netherlands, Oct. 2000.

[56] A.F.M. Marée and P. Hogeweg, "How Amoeboids Self-Organize into a Fruiting Body: Multicellular Coordination in Dictyostelium Discoideum," *Proc. Nat'l Academy of Science,* vol. 98, no. 7, pp. 3879-3883, 2001.

[57] A.F.M. Marée, A.V. Panfilov, and P. Hogeweg, "Migration and Thermotaxis of *Dictyostelium Discoideum* Slugs, A Model Study," *J. Theoretical Biology,* vol. 199, no. 3, pp. 297-309, 1999.

[58] A.F.M. Marée, A.V. Panfilov, and P. Hogeweg, "Phototaxis during the Slug Stage of *Dictyostelium Discoideum*: A Model Study," *Proc. Royal Soc. of London Series B-Biological Sciences,* vol. 266, no. 1426, pp. 1351-1360, 1999.

[59] H. Meinhardt, *Models of Biological Pattern Formation.* London: Academic Press, 1982.

[60] H. Meinhardt, "Pathways and Building Blocks: Review of "Modularity in Development and Evolution,"" *Nature,* Schlosser and Wagner, eds., vol. 430, no. 7003, p. 970, 2004.

[61] R.M.H. Merks and J.A. Glazier, "A Cell-Centered Approach to Developmental Biology," *Phys. A,* vol. 352, no. 1, pp. 113-130, 2005.

[62] R.M.H. Merks, S.A. Newman, and J.A. Glazier, "Cell-Oriented Modeling of *In Vitro* Capillary Development," *Proc. Cellular Automata: Sixth Int'l Conf. Cellular Automata for Research and Industry, ACRI 2004,* vol. 3305, pp. 425-434, Springer-Verlag, 2004.

[63] T. Miura and K. Shiota, "Tgf-beta 2 Acts as an "Activator" Molecule in Reaction-Diffusion Model and is Involved in Cell Sorting Phenomenon in Mouse Limb Micromass Culture," *Developmental Dynamics,* vol. 217, no. 3, pp. 241-249, 2000.

[64] M.Z. Moftah, S.A. Downie, N.B. Bronstein, N. Mezentseva, J. Pu, P.A. Maher, and S.A. Newman, "Ectodermal FGFs Induce Perinodular Inhibition of Limb Chondrogenesis in Vitro and in Vivo via FGF Receptor 2," *Developmental Biology,* vol. 249, no. 2, pp. 270-282, 2002.

[65] A. Mogilner and L. Edelstein-Keshet, "Regulation of Actin Dynamics in Rapidly Moving Cells: A Quantitative Analysis," *Biophysics J,* vol. 83, no. 3, pp. 1237-1258, 2002.

[66] J.C.M. Mombach, R.M.C. de Almeida, G.L. Thomas, A. Upadhyaya, and J.A. Glazier, "Bursts and Cavity Formation in *Hydra* Cell Aggregates: Experiments and Simulations," *Physica A,* vol. 297, nos. 3-4, pp. 495-508, 2001.

[67] J.C. M. Mombach and J.A. Glazier, "Single Cell Motion in Aggregates of Embryonic Cells," *Physical Rev. Letters,* vol. 76, no. 16, pp. 3032-3035, 1996.

[68] J.C. M. Mombach, D. Robert, F. Graner, G. Gillet, G.L. Thomas, M. Idiart, and J.P. Rieu, "Rounding of Aggregates of Biological Cells: Experiments and Simulations," *Physica A,* vol. 352, pp. 525-534, 2005.

[69] NEURON, Neuron Web site, 2005, http://www.neuron.yale.edu/neuron.

[70] M.E.J. Newman and G.T. Barkema, *Monte Carlo Methods in Statistical Physics.* Oxford Univ. Press, 1999.

[71] S.A. Newman and H.L. Frisch, "Dynamics of Skeletal Pattern Formation in Developing Chick Limb," *Science,* vol. 205, no. 4407, pp. 662-668, 1979.

[72] OGLE, Ogle Large-Scale Scientific Data Visualizer, 2001, http://www.cora.nwra.com/Ogle.

[73] W.B. Ouchi, J.A. Glazier, J.P. Rieu, A. Upadhyaya, and Y. Sawada, "Improving the Realism of the Cellular Potts Model in Simulation of Biological Cells," *Physica A,* vol. 329, nos. 3-4, pp. 451-458, 2003.

[74] L. Raeymaekers, "Dynamics of Boolean Networks Controlled by Biologically Meaningful Functions," *J. Theoretical Biology,* vol. 218, no. 3, pp. 331-341, 2002.

[75] R.D. Riddle, R.L. Johnson, E. Laufer, and C. Tabin, "Sonic Hedgehog Mediates the Polarizing Activity of the ZPA," *Cell,* vol. 75, no. 7, pp. 1401-1416, 1993.

[76] J.P. Rieu, A. Upadhyaya, J.A. Glazier, B.O. Noryuki, and Y. Sawada, "Diffusion and Deformations of Single *Hydra* Cells in Cellular Aggregates," *European Biophysics J.,* vol. 79, no. 4, pp. 1903-1914, 2000.

[77] N.J. Savill and P. Hogeweg, "Modelling Morphogenesis: From Single Cells to Crawling Slugs," *J. Theoretical Biology,* vol. 184, no. 3, pp. 229-235, 1997.

[78] J. Schaff, C.C. Fink, B. Slepchenko, J.H. Carson, and L.M. Loew, "A General Computational Framework for Modeling Cellular Structure and Function," *European Biophysics J.,* vol. 73, no. 3, pp. 1135-1145, 1997.

[79] A. Shalloway and J.R. Trott, *Design Patterns Explained: A New Perspective on Object-Oriented Design.* Boston, Mass.: Addison-Wesley, 2002.

[80] M.S. Steinberg, "Reconstruction of Tissues by Dissociated Cells," *Science,* vol. 141, no. 3579, pp. 401-408, 1963.

[81] M.S. Steinberg, "Goal-Directedness in Embryonic Development," *Integrative Biology,* vol. 1, pp. 49-59, 1998.

[82] M.A. Stijnman, R.H. Bisseling, and G.T. Barkema, "Partitioning 3D Space for Parallel Many-Particle Simulations," *Computer Physics Comm.,* vol. 149, no. 3, pp. 121-134, 2003.

[83] K. Takahashi, N. Ishikawa, Y. Sadamoto, S. Ohta, A. Shiozawa, F. Miyoshi, Y. Naito, Y. Nakayama, and M. Tomita, "E-CELL2: Multi-Platform E-CELL Simulation System," *Bioinformatics,* vol. 19, no. 13, pp. 1727-1729, 2003.

[84] D. Thompson, *On Growth and Form.* Cambridge Univ. Press, 1917.

[85] A. Turing, "The Chemical Basis of Morphogenesis," *Phil. Trans. Roy. Soc. London, B,* vol. 237, pp. 37-72, 1952.

[86] UNFP, *User Notes on Fortran Programming (UNFP): An Open Cooperative Practical Guide,* 2005, http://www.ibiblio.org/pub/languages/fortran/unfp.html.

[87] A. Upadhyaya, "Thermodynamic and Fluid Properties of Cells, Tissues and Membranes," PhD thesis, Univ. of Notre Dame, 2000.

[88] A. Upadhyaya, J.P. Rieu, J.A. Glazier, and Y. Sawada, "Anomalous Diffusion in Two-Dimensional *Hydra* Cell Aggregates," *Physica A,* vol. 293, nos. 3-4, pp. 549-558, 2001.

[89] Virtual Cell, National Resource for Cell Analysis and Modeling, http://www.nrcam.uchc.edu, 2005.

[90] G. von Dassow and E. Meir, "Exploring Modularity with Dynamical Models of Gene Networks," *Modularity in Development and Evolution,* G. Schlosser and G.P. Wagner, eds., pp. 244-287, 2004.

[91] Y. Wallach and V. Conrad, "On Block Parallel Methods for Solving Linear Equations," *IEEE Trans. Computers,* vol. 29, no. 5, pp. 354-359, 1980.

[92] S. Wong, "A Cursory Study of the Thermodynamics and Mechanical Properties of Monte-Carlo Simulations of the Ising Model," PhD thesis, Univ. of Notre Dame, Notre Dame, Indiana, 2005.

[93] M. Zajac, "Modeling Convergent Extension by Way of Anisotopic Differential Adhesion," PhD thesis, Univ. of Notre Dame, Notre Dame, Indiana, 2002.

[94] M. Zajac, G.L. Jones, and J.A. Glazier, "Model of Convergent Extension in Animal Morphogenesis," *Physical Rev. Letters,* vol. 85, no. 9, pp. 2022-2025, 2000.

[95] M. Zajac, G.L. Jones, and J.A. Glazier, "Simulating Convergent Extension by Way of Anisotropic Differential Adhesion," *J. Theoretical Biology,* vol. 222, no. 2, pp. 247-259, 2003.

[96] W. Zeng, G.L. Thomas, S.A. Newman, and J.A. Glazier, "A Novel Mechanism for Mesenchymal Condensation during Limb Chondrogenesis in Vitro," *Math. Modeling and Computing in Biology and Medicine: Proc. Fifth Conf. European Soc. Math. and Theoretical Biology,* pp. 80-86, 2002.

Trevor M. Cickovski received the MS degree in computer science and engineering from the University of Notre Dame. He is in his fourth year of graduate study in the Department of Computer Science and Engineering at the University of Notre Dame, Notre Dame, Indiana, directed by Dr. Izaguirre. His current research interests include domain-specific language development, stochastic simulations of biocomplexity, and software engineering.

Chengbang Huang received the MS degree in mathematics from the University of Notre Dame. He is a doctoral candidate in the Department of Computer Science and Engineering at the University of Notre Dame, Notre Dame, Indiana, directed by Dr. Izaguirre. His current research interest is the use of a multimodel framework to simulate avian limb growth.

Rajiv Chaturvedi received the PhD degree in computational fluid dynamics from the Indian Institute of Technology, Bombay, India. He is a postdoctoral research associate at the University of Notre Dame, Notre Dame, Indiana. His current research interests include computational biology and software engineering. He has been working on models of biological phenomena occurring at multiple scales, and their integration.

Tilmann Glimm received the PhD degree in mathematics from Emory University. He is a postdoctoral research associate in the Emory University Physics Department, Atlanta, Georgia, where he is working on the mathematical modeling of limb development and mesenchymal cell condensation. In general, his research interest is the analysis of nonlinear partial differential equations. He has studied at the Technical University, Berlin, Germany, and Emory University.

H. George E. Hentschel received the PhD degree in theoretical chemistry from the University of Cambridge, Cambridge, England. He is a professor of physics at Emory University, Atlanta, Georgia. His current research interests are nonlinear and biological physics.

Mark S. Alber received the PhD degree in mathematics from the University of Pennsylvania, Philadelphia, Pennsylvania. He is a professor of mathematics, concurrent professor of physics, and director of the Interdisciplinary Center for the Study of Biocomplexity (ICSB) at the University of Notre Dame, Notre Dame, Indiana. His current research interests include methods of nonlinear dynamical systems and statistical mechanics with applications in biology.

James A. Glazier received the BA degree in physics and mathematics from Harvard College, Cambridge, Massachusetts, and the PhD degree in soft condensed matter physics from the University of Chicago, Chicago, Illinois. He is a professor of physics, adjunct professor of informatics and biology, and director of the Biocomplexity Institute at Indiana University, Bloomington, Indiana. His current research interests include biophysics, development, DNA sequence analysis, neuroscience, and the mechanics of liquid foams. He is a fellow of the Institute of Physics.

Stuart A. Newman received the AB degree from Columbia University, New York, and the PhD degree in chemical physics from the University of Chicago, Chicago, Illinois. He is a professor of cell biology and anatomy at New York Medical College, Valhalla, New York. He has contributed to several scientific fields, including biophysical chemistry, developmental biology, and evolutionary theory. His current research interests include the mechanisms of vertebrate limb development, the dynamics of collagen assembly, and the evolution of morphogenesis.

Jesús A. Izaguirre received the PhD degree in computer science from the University of Illinois at Urbana-Champaign, Urbana, Illinois, in 1999. Dr. Izaguirre received a CAREER Award from the US National Science Foundation in 2001. He is an associate professor of computer science and engineering at the University of Notre Dame, Notre Dame, Indiana. His current research is on efficient methods in chemistry and biology, particularly molecular dynamics, Monte Carlo methods, cellular automata, and analysis of biological networks. He is also interested in the portable implementation of high-performance software for scientific computing. He is a member of the IEEE and the IEEE Computer Society.