Computational Modeling of Synthetic Microbial Biofilms

ARTICLE *in* ACS SYNTHETIC BIOLOGY · AUGUST 2012

Impact Factor: 4.98 · DOI: 10.1021/sb300031n · Source: PubMed

4 AUTHORS, INCLUDING:

Tim Rudge
University of Cambridge
14 PUBLICATIONS **186** CITATIONS
SEE PROFILE

Andrew Phillips
Microsoft
49 PUBLICATIONS **924** CITATIONS
SEE PROFILE

Jim Haseloff
University of Cambridge
85 PUBLICATIONS **7,529** CITATIONS
SEE PROFILE

ACS SyntheticBiology

Research Article

pubs.acs.org/synthbio

Computational Modeling of Synthetic Microbial Biofilms

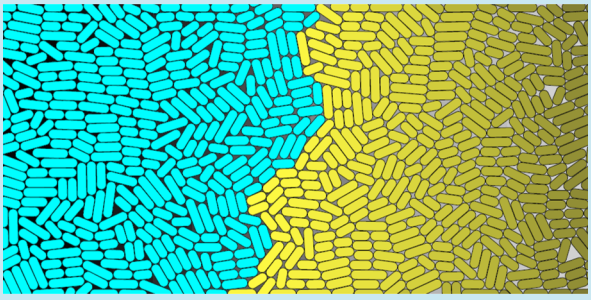
Timothy J. Rudge,[†,¶] Paul J. Steiner,[†,¶] Andrew Phillips,*[,‡] and Jim Haseloff*[,†]

[†]Department of Plant Sciences, University of Cambridge, Cambridge, U.K.

[‡]Microsoft Research, Cambridge, U.K.

Supporting Information

ABSTRACT: Microbial biofilms are complex, self-organized communities of bacteria, which employ physiological cooperation and spatial organization to increase both their metabolic efficiency and their resistance to changes in their local environment. These properties make biofilms an attractive target for engineering, particularly for the production of chemicals such as pharmaceutical ingredients or biofuels, with the potential to significantly improve yields and lower maintenance costs. Biofilms are also a major cause of persistent infection, and a better understanding of their organization could lead to new strategies for their disruption. Despite this potential, the design of synthetic biofilms remains a major challenge, due to the complex interplay between transcriptional regulation, intercellular signaling, and cell biophysics. Computational modeling could help to address this challenge by predicting the behavior of synthetic biofilms prior to their construction; however, multiscale modeling has so far not been achieved for realistic cell numbers. This paper presents a computational method for modeling synthetic microbial biofilms, which combines three-dimensional biophysical models of individual cells with models of genetic regulation and intercellular signaling. The method is implemented as a software tool (*CellModeller*), which uses parallel Graphics Processing Unit architectures to scale to more than 30,000 cells, typical of a 100 μm diameter colony, in 30 min of computation time.



KEYWORDS: *microbial, biofilm, simulation, biophysics, morphology, CellModeller*

Bacteria form self-organized communities termed biofilms, which are composed of cells embedded in a secreted extracellular matrix.[1] Cells in a biofilm differentiate phenotypically through spatially patterned gene expression[2] and can form elaborate morphological structures.[3] This behavior is coordinated by many forms of signaling, including quorum sensing, the production and sensing of diffusible small molecules[4] or peptides,[5] and the contact-based signaling of myxobacteria.[6] Downstream regulatory processes, including transcriptional regulation and biophysical interactions between individual cells, are also important.

Biofilms achieve metabolic efficiency by employing physiological cooperation similar to that observed in the tissues of multicellular organisms and are also less sensitive to changes in their local environment than planktonic cultures. Both of these properties make biofilms an attractive target for engineering. Synthetic biofilms could be engineered for the production of chemicals such as pharmaceutical ingredients or biofuels, resulting in improved yields, lower maintenance costs, and the ability to implement more complex pathways. Biofilms are also a major cause of persistent infection, and an understanding of their organization could lead to new strategies for their disruption.[7]

A key factor in the efficiency and robustness of biofilms lies in their spatial organization.[1] Biofilm initiation begins with surface attachment, followed by microcolony formation.[1] Colonies then go on to form elaborate structures, such as water channels[1] and mushroom-like stalks.[8] These structures have been examined in model systems such as *Pseudomonas aeruginosa*, using confocal microscopy at the microcolony stage, and during the development of mushroom structures.[9] Flow conditions are also known to affect the organization and morphology of biofilms and can be studied using microfluidic flow-channels.[9] Similar techniques have also been used to examine spatiotemporal dynamics of synthetic bacterial populations.[10]

Because of the dependence of morphology and patterning on cell growth and division, the design of synthetic biofilms requires the ability to predict population behaviors at single cell resolution. The diffusion limited aggregation (DLA) model was applied in early work on colony morphology, in which local depletion of nutrients gave rise to fractal morphologies.[11] However, this model does not explicitly take into account cell shape and orientation. Several studies have also used cellular simulations to model biofilm metabolism[12] or to show the effects of constraints such as bacterial growth channels on the spatial arrangements of cells depending on cell shape and size.[13−15] More recently, there have also been attempts to

Special Issue: Bio-Design Automation

Received: April 15, 2012
Published: July 11, 2012

model the interaction of such physical constraints with genetic regulation and signaling.[16,17]

Despite this progress, one of the major obstacles to designing synthetic biofilms is the number of cells in the biofilm. For a typical microcolony there can be 10^4–10^5 cells, making simulation highly computationally intensive. We have developed a rigid-body method that includes growth of cells, which results in a sparse matrix inversion problem amenable to parallel numerical solution.

The recent development of general purpose computing on Graphics Processing Units (GPUs) has enabled the simulation of large-scale parallel problems on commodity hardware, with very large numbers of parallel threads. Modeling large isogenic cell populations is ideally suited to this computational model. The *OpenCL* cross-platform framework enables implementation of parallel software that runs efficiently on both GPU and CPU architectures.[18] Furthermore, *OpenCL*'s use of run-time compilation supports flexible and dynamic simulation algorithms, which are well-suited for a design methodology.

We have implemented this method as part of the *CellModeller* software tool for multicellular modeling, using *OpenCL* for parallel computation. This parallel framework allows us to run large scale simulation of biofilms. The software includes models of biophysics, genetics, and intercellular signaling. Starting from a single cell it can simulate the development of colonies containing 30,000 individual cells in 30 min. It reproduces the main features of large scale colony morphology and can simulate realistic experimental growth conditions with fluid flows and hard or soft physical constraints. Built on the same principles as our previous multicellular methods,[19,20] *CellModeller4* allows specification of cell behavior through both rule-based and differential equation models.

RESULTS

Modeling Framework. We have developed a modular framework for the combined modeling of intracellular dynamics, intercellular signaling, and cellular biophysics. Each of these components can change the internal and external environment of a cell, and these changes can in turn propagate to other components via multiple feedback loops. For example, a cell may sense its local signal concentration and activate transcription, which could affect growth, moving the cell and changing its local signal concentration. This change in concentration could itself affect transcription, resulting in a feedback loop.

In simulations of biofilms, each cell is coupled to many others through biophysical interactions and signaling. Since growth occurs on a longer time scale than biochemical interactions, we update growth in discrete steps, and solve for the intracellular and signaling system separately. After each biophysical step, the state of each cell (position, volume etc.) is updated, and the intracellular and signaling systems are integrated forward by the appropriate time step.

Even though the simulation is highly coupled, cells of a given type are applying the same rules and differential equations to their current state. It is natural then to apply a single instruction multiple data (SIMD) approach to compute the contribution of each cell to the overall simulation in parallel.

The various components of the framework are combined in the following procedure:

1. Call an `update` function to apply user-defined rules to each cell.
2. Divide cells that have their `divideFlag` set to `True`.
3. Integrate the growth of cells forward by a chosen time step Δt, solve constraints to obtain new cell positions.
4. Integrate the species and extra-cellular signals $\vec{u}$ of each cell forward by the time step Δt.
5. Update the state variables of each cell and repeat from step 1.

Cell Biophysics. Rod-shaped bacteria maintain highly consistent forms of roughly constant radius, with growth occurring exclusively on the long axis. This shape can be approximated by a cylinder capped with hemispherical ends, called a *capsule*. In typical growth conditions, cells exhibit very little deformation so that they are well-approximated by rigid, elongating capsules. This observation led us to formulate a novel constrained rigid-body dynamics method, in which cell length is included as a degree of freedom. In the conventional rigid-body approach, the cell would be described at a given time t by its state $\vec{x}$:

$$\vec{x}(t) = (c_x, c_y, c_z, \phi_x, \phi_y, \phi_z)^T \tag{1}$$

where $(c_x,c_y,c_z)^T$ is the position of the center of mass, and $(\phi_x,\phi_y,\phi_z)^T$ is the orientation. In our scheme we include the cell length L, so that

$$\vec{x}(t) = (c_x, c_y, c_z, \phi_x, \phi_y, \phi_z, L)^T \tag{2}$$

Let us call this the *generalized position* of the cell.

Each cell grows at some rate $\dot{L}$, and for exponentially growing cells $\dot{L} \propto L$.[21] For small time periods we linearize this growth rate so that $\dot{L}$ is a constant. Note that integrating $\dot{L}$ forward in time would cause neighboring cells growing toward each other to overlap. The change in generalized position required to prevent this overlap can be formulated as a linear system:

$$\mathbf{J}\Delta\vec{x} + \vec{d} = \vec{\varepsilon} = 0 \tag{3}$$

where $\vec{d}$ are the overlap distances for pairs of cells, the matrix $\mathbf{J}$ $(n_{\text{contacts}} \times n_{\text{cells}})$ encodes the change in overlap for a given change in position $(\Delta\vec{x})$, and $\vec{\varepsilon}$ is the resulting overlap after the position change (see Supporting Information for details). We include plane constraints in this system as additional rows, with the entry in $\vec{d}$ the penetration distance into the plane.

Equation 3 is an ill-posed underdetermined linear system, and so we compute the regularized least-squares solution by minimizing $\|\vec{\varepsilon}\|^2$. In order to model soft constraints such as an agarose substrate, we apply a diagonal weight matrix $\|\mathbf{D}\vec{\varepsilon}\|^2$, with $D_{ij} < 1$ for soft constraints, $D_{ij} = 1$ otherwise. This is analogous to elastic constraints with relative stiffness matrix $\mathbf{K} = \mathbf{D}^2$

In the low Reynolds number regime appropriate to bacteria, viscous drag dominates inertia, and cells will move by a distance proportional to the impulse applied to them. Thus, for given $\vec{x}$ and $\vec{d}$ we solve for the impulse $\Delta\vec{p}$ that must be applied to each cell to satisfy the constraint eq 3:

$$\mathbf{J}\mathbf{M}^{-1}\Delta\vec{p} + \vec{d} = \vec{\varepsilon} = 0 \tag{4}$$

where $\mathbf{M}$ is a matrix containing the viscous drag coefficients, including a viscocity term accounting for the cell's resistance to change in length. We regularize the least-squares solution of 4 by minimizing $\Delta\vec{p}^T\mathbf{M}^{-1}\Delta\vec{p}$. (See Supporting Information for a full mathematical description.)

The matrices $\mathbf{J}$ and $\mathbf{M}$ are sparse and structured so that eq 4 is well suited to solution by a matrix free iterative method. In

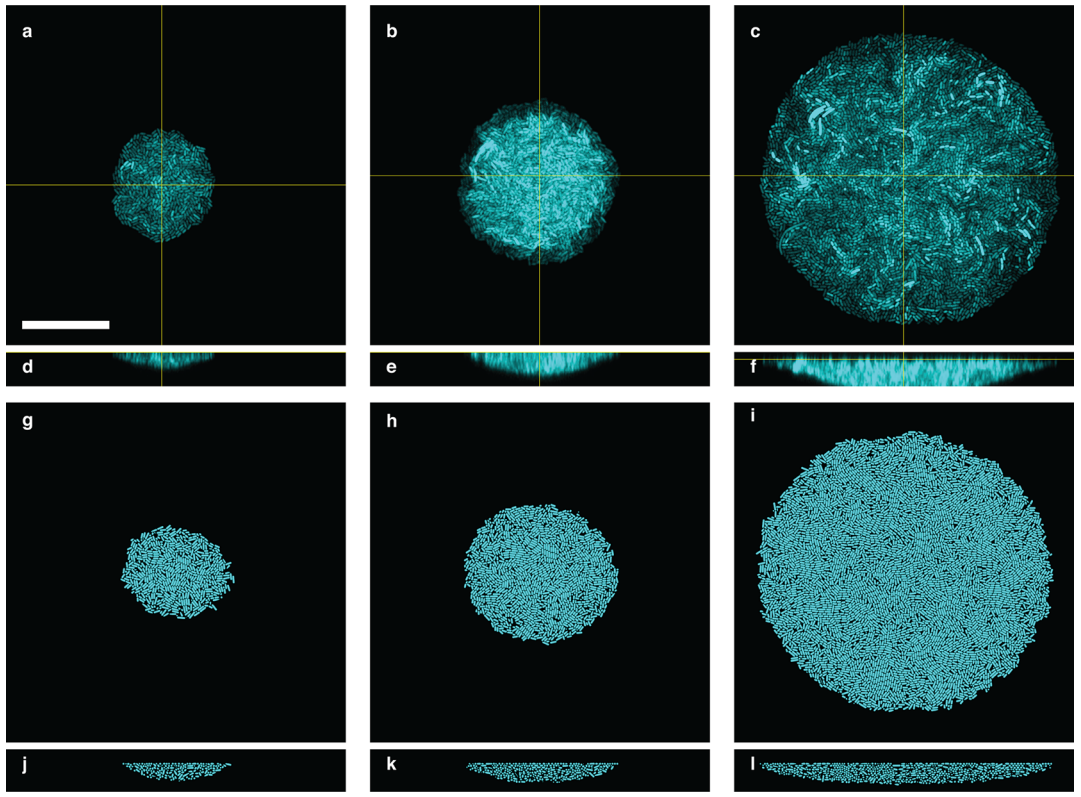


Figure 1. Confocal images of *E. coli* colonies compared with simulation results. (a–f) *E. coli* expressing ECFP on a medium copy plasmid (p15a origin), grown at room temperature. Images taken 2 h apart, beginning 6 h after inoculation. Yellow lines in panels a–c show the location of XZ slices shown in panels d–f. (g–l) Simulation of colony initiated from single cell in microscopy conditions. Final cell number ~32,000. Total computation time around 30 min. Scale bar in panel a: 30 μm. All images at the same scale.

this approach, the full matrix is not stored, but matrix-vector products are computed as needed to update the iteration. GPU solution of such sparse matrix linear systems has been shown to give significant speed up.[22] In our system these matrix-vector products are sums over each cell's contacts and can therefore be computed in parallel.

We implemented this implicit matrix multiplication calculation in *OpenCL* and used it to iterate a conjugate gradient solver. We also used *OpenCL* to efficiently find pairs of contacting cells with a spatial grid-based approach, where collisions need only be tested for cells within a neighborhood on the grid. Full details of these algorithms and their implementation can be found in Supporting Information.

At cell division, each dividing cell is replaced with two cells of half the length of the parent, such that they occupy the same space. In order to simulate imperfections in cell shape and alignment, a small amount of noise (0.1%) is the added to the direction vector of each daughter cell.

Microcolony Morphology. We used our rigid-body method to compare model simulations of microcolony morphology with experimental results. Early stage biofilms form microcolonies after attachment of a single cell to a surface. This situation can be produced experimentally by initiating colonies from dilute liquid culture on an agarose pad with a coverslip placed on top. In this way single cells expressing fluorescent protein can be isolated, and using confocal microscopy their development into microcolonies can be tracked in 3 dimensions over time. Figure 1 shows a typical colony developing over around 18 h at room temperature. The form is circular, with a dome shaped profile extending into the agarose. Alignment of cells is typically in-plane, with some cells forced into near vertical orientations.

Our model reproduces this form, including the orientation of cells. Figure 2 shows arrangements of cells, with vertical cells

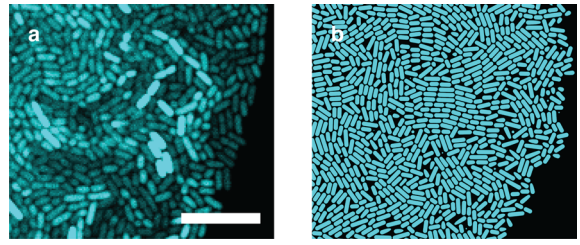


Figure 2. Cell arrangements in (a) an *E. coli* colony and (b) a simulation. Each image is on the same scale. Scale bar: 10 μm.

appearing as circles. To approximate the experimentally measured variation in the length of newborn *E. coli* cells,[23,24] simulations were based on a simple rule that cells grow at the same rate and divide when their length reaches a uniformly distributed threshold in the range of 3.5–4.5 μm. We simulated the microscopy conditions by introducing two plane constraints. A soft plane constraint below the cells modeled the agarose pad on which they were grown. A hard plane constraint above the cells modeled the coverslip.

The final colony contains approximately 32,000 cells and was computed in around 30 min, showing that our parallel method allows simulations of systems large enough to demonstrate colony-stage biofilm morphology. The simulation time required to obtain a cell colony of a given size, starting from a single cell,

is shown in Figure 3. Simulation time increases with the number of cells in the colony due to portions of the algorithm

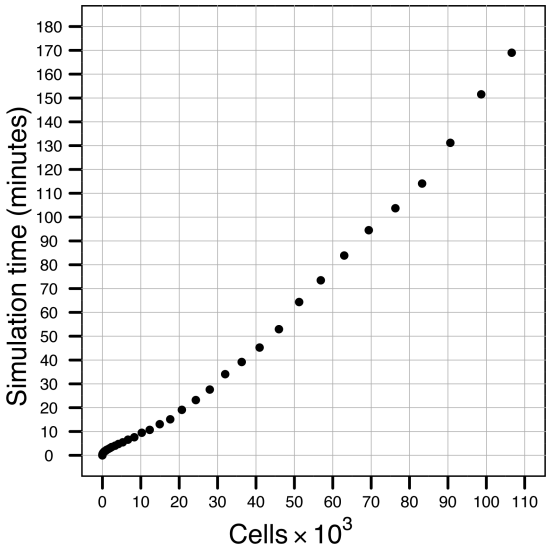


Figure 3. Simulation time for cell colonies of different sizes. Each point on the graph denotes the time in minutes for a cell colony of a given size to be simulated by CellModeller, starting from a single cell.

being implemented on a CPU, which incurs an overhead that increases with colony size. Despite this overhead, the GPU implementation of matrix vector products allows for significant speedup.[22] For comparison, a recent study on simulation of spherical cell populations reported computation times of around 40 h to simulate 10^5 cells,[17] compared to around 2.5 h for our method (see Figure 3).

Rule-Based Cell Behavior. Although there are numerous and complex regulatory mechanisms that determine cell behavior, in many cases we do not need to model these mechanisms explicitly. Instead, it is often sufficient to consider empirically derived rules for these processes. In our method we allow the definition of such rules in the *Python* programming language, by defining a suitable `update` function. For example, the following update function applies the rule that all cells grow at a relative rate of 0.035 min^{-1} (equivalent to a doubling time of 20 min), and divide when they reach a volume of 3 μm^3.

```
def update(cells):
    for cell in cells:
        cell.growthRate = 0.035
    if cell.volume > 3:
        cell.divideFlag = True
```

This update is called at each simulation step with the current list of cells. It can be modified at run-time and immediately applied, meaning that models can be interactively developed while they are running.

Each cell contains a standard set of *state variables* listed in Table 1, and because *Python* is flexibly typed, the user can also create new variables. More complicated rules can be developed in a number of ways, including introducing random variation, partitioning into cell types, and using the outcome of differential equation models.

When a cell's `divideFlag` is equal to `True` it is divided into two daughter cells. By default each cell inherits the state variable values of its parent. The user also can specify a

Table 1. Each Cell Is Described by the Following *state variables*, to Which Rules May Be Applied

name	usage	meaning
`pos`	read	centre of mass of cell
`dir`	read	orientation of cell
`length`	read	length (μm)
`radius`	read	radius (μm)
`volume`	read	volume (μm^3)
`area`	read	surface area (μm^2)
`species`	read	list of internal species concentrations
`signals`	read	list of local signal concentrations
`cellType`	read/write	integer cell type identifier
`growthRate`	read/write	relative growth rate of cell (min^{-1})
`divideFlag`	read/write	triggers cell division when `True`
*	read/write	User-defined variables

`divide(parent, daughter1, daughter2)` function, which can be used to implement specific models of division. For example, the random partitioning of a molecular species between two daughter cells following cell division could be modeled as follows:

```
def divide(parent, daughter1, daughter2):
  # Daughter1 inherits fraction r1 of
  # parent concentration, and daughter2
  # inherits the remainder
  r1 = random.uniform(0.0, 1.0)
  r2 = 1-r1
  daughter1.species[0] = parent.species[0]*r1
  daughter2.species[0] = parent.species[0]*r2
```

The flexibility of the cell state structure, including allowing user-defined variables, means that a broad range of cell behavior rules can be encoded, such as differentiation and inheritance.

Programmed Growth. We used our rule-based method to model programmed growth of a microcolony. One of the goals of engineering biofilms is to generate novel morphologies by spatially organized growth. This growth would be organized by patterns of gene expression triggering growth effectors, which could be metabolic processes. As a first step we explored the effects of growth rate on colony morphology by imposing patterns of gene expression. Such imposed patterns have already been achieved with light-sensing phytochromes.[25,26]

Although we do not have growth effectors or a full design for programmed induction, one of the powerful aspects of rule-based modeling is that we can still explore the possible effects of programmed morphology. Starting from a single cell, we use the `update` function to specify a rule that only cells within 10 μm of the rightmost edge of the colony are able to grow. In practice this could be imposed with a moving mask of exposure to the appropriate wavelength of light for the phytochrome, with only exposed cells triggered to grow. The result (Figure 4 and Supplementary Video) shows meristem-like properties, with the growing cells propelled to the right.

Intracellular Dynamics and Signaling. Our method can be used to model the operation of synthetic constructs, such as transcription regulation circuits, in a cellular biofilm context. Depending on the construct in question, different levels of abstraction may be appropriate. In some cases the activity of cells may be abstracted to simple rules, for example, to study the effects of growth rate on colony morphology. In other cases a more detailed model is required, for example, to study the behavior of transcription networks in cell colonies.

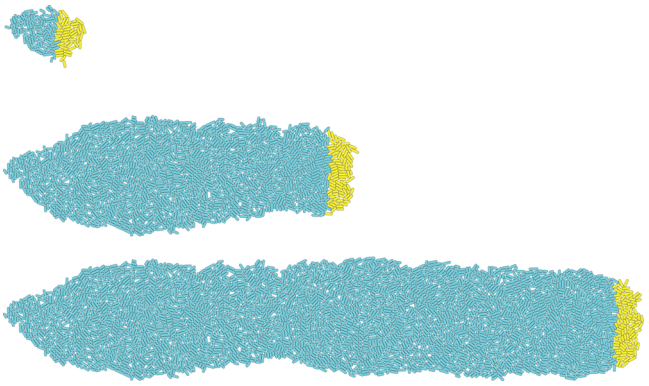


Figure 4. Three snapshots of a simulation of controlled growth, where only cells within 10 μm of the rightmost edge of the colony are able to grow (yellow). This could be achieved by optical induction of a phytochrome system linked to metabolic control genes. Total computation time was 28 min. (See Supporting Information for a video of this simulation.).

Intracellular dynamics are commonly approximated by ordinary differential equations, which have been shown to accurately reproduce experimental observations in a broad range of synthetic systems. Examples include transcriptional oscillators,[27] quorum-sensing systems with predator–prey interactions,[28] and RNA-regulated genetic devices.[29] Despite this success, the limitations of ordinary differential equations are well-documented, particularly with regards to variability in gene expression.[30] While variability between cells can be simulated in CellModeller by introducing randomness at specific events, such as the partitioning of a molecular species between daughter cells during cell division, variability at the level of gene expression would be more accurately modeled using stochastic simulation methods. Although such methods are highly computationally intensive, various parallel algorithms for their implementation have been proposed (see ref 31 for example). Here we suggest that the stochastic reactions occurring in each cell could be simulated in parallel, rather than realizing a single stochastic trajectory. We leave the implementation of this proposal for future work.

For a given synthetic construct, we represent the processes that require detailed modeling as a system of differential equations, with each equation describing the rate of change of a species. In general, and most commonly for genetic circuits, this system is nonlinear, where the rate of change of the vector of species $\vec{u}$ is of the form

$$\frac{d\vec{u}}{dt} = f(\vec{u}) \tag{5}$$

Our approach is to solve this general case, with the function f specified by the user. This is computationally intensive, and we use the *OpenCL* parallel programming language to compute $f(u_i)$ for each cell i in parallel. The user must specify simple *OpenCL* code to define the rate of change of each species.

Cell signaling is a key part of multicellular organization. In the biofilm mode of growth, cells communicate via quorum sensing ligands that diffuse through the biofilm and medium, after being secreted from the cell. Depending on the environment in which the biofilm is growing, there may also be other transport processes, such as bulk flow or advection. We include such signaling through the medium in our algorithm with a general linear transport operator $\mathbf{T}$:

$$\frac{d\vec{u}}{dt} = \mathbf{T}[\vec{u}] + f(\vec{u}) \tag{6}$$

where $\vec{u}$ is now composed of some species that are within a cell (and are not transported) and those outside the cell, which are subject to the operator $\mathbf{T}$. For example, in the case of diffusion $\mathbf{T} \equiv \mathbf{K}\nabla^2$, where $\mathbf{K}$ is a diagonal matrix of diffusion coefficients for each species, and for cell autonomous species the corresponding element of $\mathbf{K}$ is zero.

We discretize this system on a regular 3-dimensional grid for species in the medium and separate variables representing cell-autonomous species. Cell positions are interpolated linearly in the spatial grid, and each cell can see its local signal concentration (see Table 1). The user writes the function $f(u)$ for each cell including, for example, importing or exporting signal and downstream transcriptional regulation. Table 2 shows example function definitions for a simple model of the Lux quorum sensing system.

Table 2. Specification of Differential Equations for *OpenCL* Solver[a]

```
def sigRateCL():
    return '''
    const float D1 = 1.f;

    float AHL = signals[0];
    float AHLi = species[1];

    rates[0] = -D1*(AHL-AHLi)*area/gridVolume;
    '''

def specRateCL():
    return '''
    const float D1 = 1.f;
    const float d1 = 1e-3;
    const float k1 = 1.f;
    const float k2 = 1.f;
    const float g1 = 1e-1;
    const float k3 = 1.f;
    const float k4 = 1.f;

    float AHL = signals[0];
    float LuxI = species[0];
    float AHLi = species[1];

    rates[0] = d2 + k3*AHLi*AHLi/(k4+AHLi*AHLi)
             - g1*LuxI;
    rates[1] = k1*LuxI/(k2+LuxI)
             + D1*(AHL-AHLi)*area/volume;
    '''
```

[a]Internal species levels and local signal concentrations, as well as cell surface area and volume, are available to use in the expressions. Here, in `sigRateCL` internal AHL is exported into the medium via the membrane (hence surface area term), and concentrations must be scaled for the change in volume. In `specRateCL`, LuxI production is induced by AHL, and internal AHL is synthesized from LuxI. Function definitions are returned as strings, which are then compiled at run-time into *OpenCL*.

We solve the resulting system of nonlinear partial differential equations using a modified Crank–Nicholson method (see Supporting Information). Our method solves the update step numerically, meaning we can define any linear transport operator in a modular fashion, such as adding an advection or bulk flow term in direction $\hat{\vec{n}}$:

$$\mathbf{T} \equiv \mathbf{K}\nabla^2 + \mathbf{C}\hat{\vec{n}}\cdot\nabla \tag{7}$$

dx.doi.org/10.1021/sb300031n | *ACS Synth. Biol.* 2012, 1, 345–352

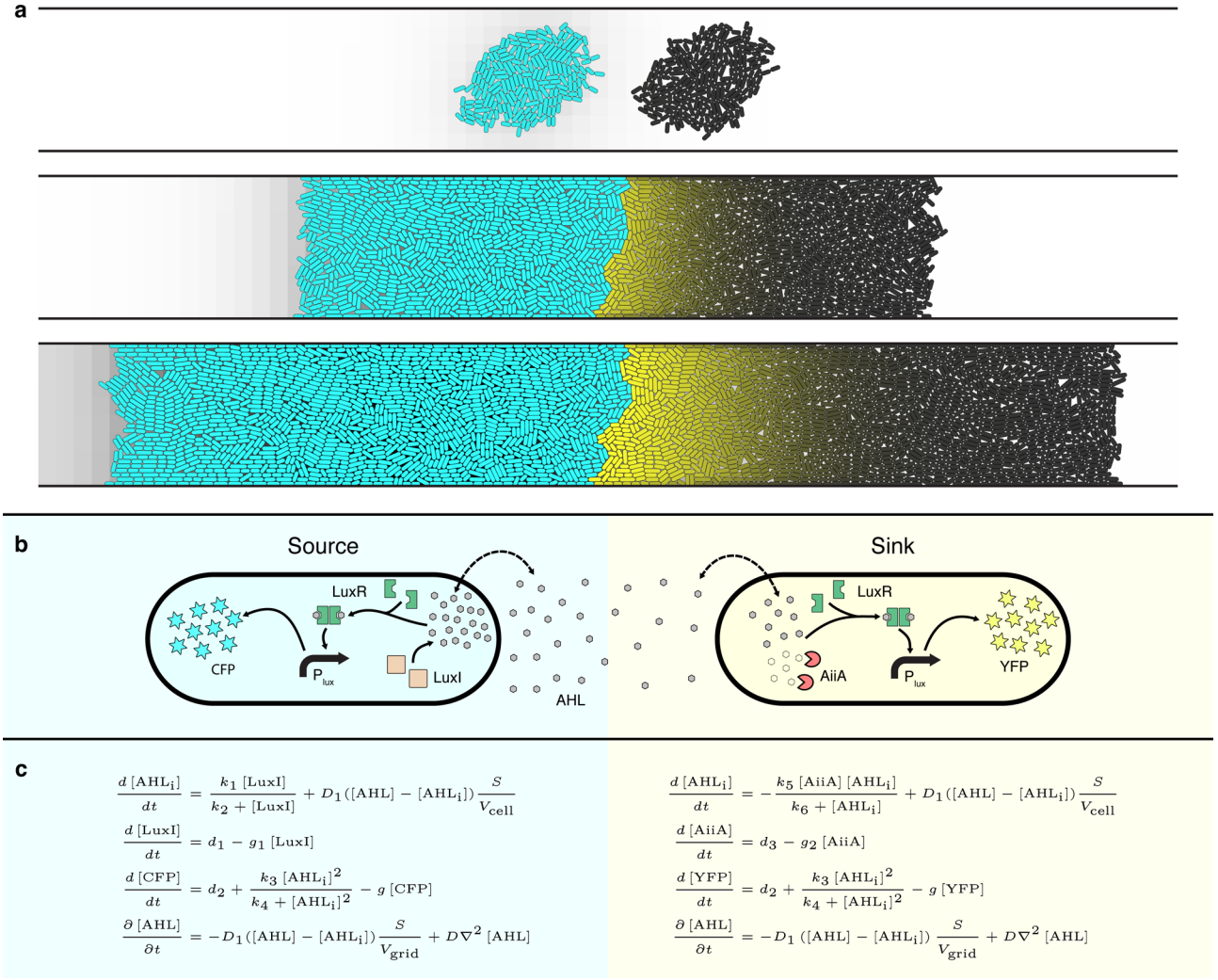


Figure 5. Detection of the boundary between two populations of cells. Both populations sense the presence of the signaling molecule. The "source" cells (cyan) constitutively produce the signal, and the "sink" cells (yellow) constitutively degrade it. Because of degradation, only the sink cells on the boundary of the two populations detect signal. (a) Three snapshots of a simulation of the domain boundary detector. Cells are constrained to a single plane in a microfluidic channel. Intensity of color indicates fluorescent protein levels in individual cells. Background indicates AHL level in the environment with white lowest and black highest. (b) Diagram of the simulated system. AHL (gray hexagons) acts as the signaling molecule, which passively diffuses across the cell membrane. LuxI synthesizes AHL, and AiiA degrades it. LuxR binds internal AHL and induces expression of a fluorescent reporter (CFP for sources or YFP for sinks). (c) The system of differential equations used to implement the simulation in *CellModeller*. Total computation time 20 min. (See Supporting Information for a video of this simulation.)

where **C** is the matrix of flow rates for each species.

Domain Boundary Detection. We used this approach to model domain boundary detection, by examining the growth of two communicating cell populations in a microfluidic channel. Maintenance and refinement of boundaries between such cohorts of cells is known to be critical in developmental systems[32] and thus also important for engineering multicellular behaviors. Microfluidic growth channels have been used experimentally to study bacterial biophysics[15] and to observe synthetic genetic circuits.[10] Like the system constructed by Tabor et al.,[25] this system is designed to detect an edge between cell populations by sensing a diffusing signal, but it additionally incorporates degradation of the signaling molecule to decrease the width of the edge detected. One population, the source, constitutively produces LuxI, an intracellular enzyme that synthesizes the signaling molecule AHL. The other population, the sink, constitutively produces the intracellular enzyme AiiA that degrades AHL. Both populations express LuxR, a transcription factor that activates transcription from the lux promoter in the presence of AHL and respond to the presence of AHL by producing a fluorescent protein (CFP for source, YFP for sink). Figure 5 gives more detail.

The result is a band of high YFP expression at the border of the two domains. Constrained growth in the channel forces cells to grow along its length, and the marked border appears perpendicular. The irregular boundary between the two domains shows the effect of individual cell geometry. Such irregular boundaries are sharpened in developmental systems by, for example, interdomain signaling.[32] Our model could provide a framework for designing synthetic sharpening mechanisms of this kind.

DISCUSSION

In this paper we have presented a method for the simulation of biofilm-scale bacterial populations, together with an efficient software implementation of this method. The primary

dx.doi.org/10.1021/sb300031n | *ACS Synth. Biol.* 2012, 1, 345–352

difficulties in simulating large bacterial populations are (i) numerical stability of the solution to the simulated system, which we solve by a novel adaptation of rigid-body dynamics, and (ii) speed of simulation, which we solve by implementing our method in a highly parallel fashion using *OpenCL* and GPU architectures. Our software reproduces the morphology of actual bacterial colonies and can simulate varied experimental conditions such as growth in a microfluidic channel and optically controlled gene expression.

Our method allows models of biophysics, intracellular dynamics, and intercellular signaling to be programmed via discrete rules and systems of differential equations. Although these models are programmed manually at present, we are currently extending our framework to allow models to be imported using a standard interchange format, the Systems Biology Markup Language (SBML).[33] Future work will also involve a close integration of our framework with the Genetic Engineering of Cells language (GEC),[34] so that dynamic models of microbial cells can be automatically generated from a high-level system design. Such integration would provide increased automation for the design of synthetic biofilms, by allowing computational models of cell behavior to be derived from a design expressed as a composition of characterized genetic parts. The construction of candidate designs that exhibit the desired behavior in simulations would then consist of assembling the given parts, providing a close link between design and implementation.

The emergent properties exhibited by thousands of growing, signaling, and responding bacterial cells are central to the rational design of synthetic biofilms. Such emergent behavior is extremely difficult to predict and will require the development of realistic, scalable simulation methods. We have presented a first attempt at developing such a method and demonstrated its scalability and flexibility for modeling the emergent behavior of multicellular synthetic biological systems.

METHODS

Computation. *CellModeller4* is our software framework for multicellular modeling. It incorporates the model presented here, as well as biophysical models of plant cells[19,20] in a modular fashion. Further information and downloads can be found at www.cellmodeller.org. The software was written in *Python* and *OpenCL* using the packages *pyopencl*,[35] *Numpy*,[36] and *Scipy*.[37] Simulations were performed on a Hewlett-Packard Z800 workstation with an NVIDIA Quadro FX5800 graphics card. Using *OpenCL* means that the software will run on a large range of GPU and CPU architectures, and the *Python* implementation is cross-platform with respect to operating systems.

Microbial Cultures. *E. coli* strain *E Cloni 10G* (Invitrogen) was transformed with plasmid pSB3K3 from the Registry of Standard Biological Parts.[38] The insert consisted of constitutive promoter BBa_J23101, ribosome binding site BBa_0034 followed by ECFP coding sequence (BBa_E0020) and transcriptional terminator BBa_0015. Cultures were grown in M9 minimal medium supplemented with 0.4% w/v glucose, 0.2% w/v casamino acids, and 50 mg/mL kanamycin to OD_{600} of approximately 0.1. These cultures were then diluted by 10^3, and 10 μL was placed on a pad of 1.5% w/v agarose in selective medium. The pads were prepared on microscope slides as described by de Jong et al.[39] with multiple frames to increase agarose volume.

Microscopy. Prepared slides were grown at room temperature and imaged every 2 h, for a total of 18 h. Imaging was performed with a Leica SP5 laser scanning confocal microscope in upright configuration, using a 40X Plan Apo NA 1.25 oil immersion objective to take 1024 × 1024 × 20 image stacks at each time point. ECFP (emission peak 434 nm) was excited with the 458 nm line of an argon ion laser.

ASSOCIATED CONTENT

Supporting Information

Details of the mathematical model and its implementation. This material is available free of charge via the Internet at http://pubs.acs.org.

AUTHOR INFORMATION

Corresponding Author

*E-mail: andrew.phillips@microsoft.com; jh295@cam.ac.uk.

Author Contributions

¶These authors contributed equally to this work.

Notes

The authors declare no competing financial interest.

ACKNOWLEDGMENTS

T.J.R. is supported by a Microsoft Research studentship. P.J.S. is supported by a Cambridge International Scholarship. The authors thank Fernán Federici and James Brown for plasmid DNA and Michael Pedersen for helpful discussions. T.J.R. would like to thank Fernán Federici and Paul Grant for help with microscopy.

REFERENCES

(1) Costerton, J. W., Lewandowski, Z., Caldwell, D. E., Korber, D. R., and Lappin-Scott, H. M. (1995) Microbial biofilms. *Annu. Rev. Microbiol. 49*, 711−745.

(2) McLoon, A. L., Kolodkin-Gal, I., Rubinstein, S. M., Kolter, R., and Losick, R. (2011) Spatial regulation of histidine kinases governing biofilm formation in Bacillus subtilis. *J. Bacteriol. 193*, 679−685.

(3) Branda, S. S., González-Pastor, J. E., Ben-Yehuda, S., Losick, R., and Kolter, R. (2001) Fruiting body formation by Bacillus subtilis. *Proc. Natl. Acad. Sci. U.S.A. 98*, 11621−11626.

(4) Fuqua, C., Winans, S., and Greenberg, E. (1996) Census and consensus in bacterial ecosystems: The LuxR-LuxI family of quorum-sensing transcriptional regulators. *Annu. Rev. Microbiol. 50*, 727−751.

(5) Kleerebezem, M., Quadri, L. E. N., Kuipers, O. P., and deVos, W. M. (1997) Quorum sensing by peptide pheromones and two-component signal-transduction systems in Gram-positive bacteria. *Mol. Microbiol. 24*, 895−904.

(6) Kaiser, D. (2004) Signaling in Myxobacteria. *Annu. Rev. Microbiol. 58*, 75−98.

(7) Costerton, J. W., Stewart, P. S., and Greenberg, E. P. (1999) Bacterial biofilms: A common cause of persistent infections. *Science 284*, 1318−1322.

(8) Chiang, P., and Burrows, L. L. (2003) Biofilm formation by hyperpiliated mutants of Pseudomonas aeruginosa. *J. Bacteriol. 185*, 2374−2378.

(9) Pamp, S. J.; Sternberg, C.; Tolker-Nielsen, T. Insight into the microbial multicellular lifestyle via flow-cell technology and confocal microscopy. *Cytometry, Part A* 2009, *75A*, 75A, 90, 90−103, 103.

(10) Danino, T., Mondragón-Palomino, O., Tsimring, L., and Hasty, J. (2010) A synchronized quorum of genetic clocks. *Nature 463*, 326−330.

(11) Matsushita, M., and Fujikawa, H. (1990) Diffusion-limited growth in bacterial colony formation. *Phys. A (Amsterdam, Neth.) 168*, 498−506.

(12) Kreft, J. U., Booth, G., and Wimpenny, J. W. (1998) BacSim, a simulator for individual-based modelling of bacterial colony growth. *Microbiology (Reading, U.K.) 144* (Pt 12), 3275−3287.
(13) Cho, H., Jönsson, H., Campbell, K., Melke, P., Williams, J. W., Jedynak, B., Stevens, A. M., Groisman, A., and Levchenko, A. (2007) Self-organization in high-density bacterial colonies: Efficient crowd control. *PLoS Biol. 5*, e302.
(14) Volfson, D., Cookson, S., Hasty, J., and Tsimring, L. S. (2008) Biomechanical ordering of dense cell populations. *Proc. Natl. Acad. Sci. U.S.A. 105*, 15346−15351.
(15) Boyer, D., Mather, W., Mondragón-Palomino, O., Orozco-Fuentes, S., Danino, T., Hasty, J., and Tsimring, L. S. (2011) Buckling instability in ordered bacterial colonies. *Phys. Biol. 8*, 026008.
(16) Klavins, E. gro: The cell programming language. http://depts.washington.edu/soslab/gro/.
(17) Hoehme, S., and Drasdo, D. (2010) A cell-based simulation software for multi-cellular systems. *Bioinformatics 26*, 2641−2642.
(18) Khronos OpenCL Working Group (2008) *The OpenCL Specification, version 1.0.29*.
(19) Rudge, T., and Haseloff, J. (2005) in *Advances in Artificial Life*, (Hutchison, D., Kanade, T., Kittler, J., Kleinberg, J. M., Mattern, F., Mitchell, J. C., Naor, M., Nierstrasz, O., Pandu Rangan, C., Steffen, B., Sudan, M., Terzopoulos, D., Tygar, D., Vardi, M. Y., Weikum, G. et al., Eds.) Vol. *3630*; pp 78−87, Springer, Berlin, Heidelberg.
(20) Dupuy, L., Mackenzie, J., Rudge, T., and Haseloff, J. (2008) A system for modelling cell−cell interactions during plant morphogenesis. *Ann. Bot. (Oxford, U.K.) 101*, 1255−1265.
(21) Cooper, S. (1991) Synthesis of the cell surface during the division cycle of rod-shaped, Gram-negative bacteria. *Microbiol. Rev. 55*, 649−674.
(22) Bolz, J., Farmer, I., Grinspun, E., and Schröoder, P. (2003) Sparse matrix solvers on the GPU: conjugate gradients and multigrid. *ACM Trans. Graph. 22*, 917−924.
(23) Cullum, J., and Vicente, M. (1978) Cell growth and length distribution in Escherichia coli. *J. Bacteriol. 134*, 330−337.
(24) Reshes, G., Vanounou, S., Fishov, I., and Feingold, M. (2008) Timing the start of division in E. coli: a single-cell study. *Phys. Biol. 5*, 046001.
(25) Tabor, J. J., Salis, H. M., Simpson, Z. B., Chevalier, A. A., Levskaya, A., Marcotte, E. M., Voigt, C. A., and Ellington, A. D. (2009) A synthetic genetic edge detection program. *Cell 137*, 1272−1281.
(26) Tabor, J. J., Levskaya, A., and Voigt, C. A. (2011) Multichromatic control of gene expression in Escherichia coli. *J. Mol. Biol. 405*, 315−324.
(27) Elowitz, M. B., and Leibler, S. (2000) A synthetic oscillatory network of transcriptional regulators. *Nature 403*, 335−338.
(28) Balagaddé, F. K., Song, H., Ozaki, J., Collins, C. H., Barnet, M., Arnold, F. H., Quake, S. R., and You, L. (2008) A synthetic Escherichia coli predator-prey ecosystem. *Mol. Syst. Biol. 4*, 187.
(29) Carothers, J. M., Goler, J. A., Juminaga, D., and Keasling, J. D. (2011) Model-driven engineering of RNA devices to quantitatively program gene expression. *Science 334*, 1716−1719.
(30) Elowitz, M. B., Levine, A. J., Siggia, E. D., and Swain, P. S. (2002) Stochastic gene expression in a single cell. *Science 297*, 1183−1186.
(31) Dematté, L. and Mazza, T. (2008) in *Computational Methods in Systems Biology* (Hutchison, D., Kanade, T., Kittler, J., Kleinberg, J. M., Mattern, F., Mitchell, J. C., Naor, M., Nierstrasz, O., Pandu Rangan, C., Steffen, B., Sudan, M., Terzopoulos, D., Tygar, D., Vardi, M. Y., Weikum, G. et al., Eds.) Vol. *5307*; pp 191−210, Springer, Berlin, Heidelberg.
(32) Dahmann, C., Oates, A. C., and Brand, M. (2011) Boundary formation and maintenance in tissue development. *Nat. Rev. Genet. 12*, 43−55.
(33) Hucka, M., Finney, A., Sauro, H. M., Bolouri, H., Doyle, J. C., Kitano, H., Arkin, A. P., Bornstein, B. J., Bray, D., Cornish-Bowden, A., Cuellar, A. A., Dronov, S., Gilles, E. D., Ginkel, M., Gor, V., et al. (2003) The Systems Biology Markup Language (SBML): A medium for representation and exchange of biochemical network models. *Bioinformatics 19*, 524−531.
(34) Pedersen, M., and Phillips, A. (2009) Towards programming languages for genetic engineering of living cells. *J. R. Soc., Interface 6*, S437−S450.
(35) Klöckner, A., Pinto, N., Lee, Y., Catanzaro, B., Ivanov, P., and Fasih, A. (2012) PyCUDA and PyOpenCL: A scripting-based approach to GPU run-time code generation. *Parallel Computing 38*, 157−174.
(36) Oliphant, T. E. (2006) *Guide to NumPy*.
(37) http://www.scipy.org/.
(38) http://partsregistry.org.
(39) de Jong, I. G., Beilharz, K., Kuipers, O. P., and Veening, J.-W. (2011) Live cell imaging of Bacillus subtilis and Streptococcus pneumoniae using automated time-lapse microscopy. *J. Vis. Exp.*, 3145.