A rigid body framework for multicellular modeling

Phillip J. Brown1✉, J. Edward F. Green1✉, Benjamin J. Binder1✉ and James M. Osborne2✉

Off-lattice models are a well-established approach in multicellular modeling, where cells are represented as points that are free to move in space. The representation of cells as point objects is useful in a wide range of settings, particularly when large populations are involved; however, a purely point-based representation is not naturally equipped to deal with objects that have length, such as cell boundaries or external membranes. Here we introduce an off-lattice modeling framework that exploits rigid body mechanics to represent objects using a collection of conjoined one-dimensional edges in a viscosity-dominated system. This framework can be used to represent cells as free moving polygons, to allow epithelial layers to smoothly interact with themselves, to model rod-shaped cells such as bacteria and to robustly represent membranes. We demonstrate that this approach offers solutions to the problems that limit the scope of current off-lattice multicellular models.

Multicellular modeling is a technique for representing tissues as a collection of individual agents. It can be used to investigate anything from embryogenesis to crypt homeostasis. The agents can be represented by objects on a fixed lattice, or by an off-lattice approach using points that are free to move in space. In 1981, Odell and colleagues demonstrated the ability of off-lattice approaches to investigate morphogenesis caused by the mechanics of cell–cell interactions. Since then they have been used in a vast array of modeling problems and developed into several unique branches. Among them are center-based models, in which forces on a cell are applied at a single point (for instance, overlapping spheres and tessellation models), and multi-node models (such as the vertex model and the subcellular element method) where cells are represented by a collection of points. These methods, in a broad sense, are built in a node-based framework, as they calculate motion directly by using forces applied to nodes.

Motion in a node-based model is routinely realized mechanically (except by Merks and co-workers, who use random movement to reduce total energy via metropolis sampling). The net force \( \mathbf{F}_i \) applied to each node \( i \) is calculated and then its movement is determined via equations of motion. Often in cell-based modeling, inertial forces are assumed to be negligible compared with viscous drag forces, meaning the motion can be described by the first-order differential equations

\[
\eta_i \frac{d\mathbf{r}_i}{dt} = \mathbf{F}_i, \quad i \in 1...N,
\]

where \( \mathbf{r}_i \) is the position of the node and \( \eta_i \) its drag coefficient. This is solved numerically for each node to determine its new position in space, often using the forward Euler method. The net force \( \mathbf{F}_i \) can be determined either by directly calculating forces (perhaps via a spring law) or by energy methods whereby the force is the result of energy potentials. Depending on the model, other steps to reposi-
tion the nodes come after this, for instance, with cell division and death. Notably, the vertex model performs various kinds of swaps where cells are represented by a collection of edges that can interact with their simulated environment (for example, other cells) through physically realistic forces. We explore several applications of the rigid body framework and demonstrate how they can reproduce established results and overcome existing issues. These applications are: cells modeled as separated polygons used to model a tumor spheroid; a contiguous ring of rectangular cells used to represent an epithelial monolayer; bacterial cells modeled by rods growing in a restricted domain; and a membrane modeled as a ring of edges. Finally, we discuss the power of the framework, its potential applications in cell-based modeling, and highlight areas for future development. A detailed description of the underlying theory is provided in Methods.

Results

A rigid body framework for multicellular modeling. The fundamental unit of our modeling framework is a one-dimensional edge. This means we need to consider where on the edge forces are applied as well as how the edge rotates.

We define an edge by a linked pair of nodes, which specifies its endpoints, and a line joining the endpoints, which represents its
Force applied to a rigid body (Fig. 1c(i)) can be replaced with a force at the center of drag and a moment (ii). This is then used to calculate the ghost rigid body displacement that an isolated edge would experience in a given time step (gray edge in (iii)). From this, we reverse engineer the equivalent forces that would produce the same resultant displacement of the nodes when they are treated independently of the edge (iv).

Interior. Forces can be applied directly to the endpoints, or at some point along the interior, and these forces will generally be caused by interactions with other edges in the model. We limit our scope to only consider interactions among endpoints (which we term node–node interactions), and between endpoints and interiors (node–edge interactions). It is possible to specify interactions between interiors (edge–edge), but we leave this for future development.

Figure 1a,b shows how we determine where interaction forces are applied. A circle is centered on an endpoint of an adjacent edge, and its radius extended until it touches the edge of interest. If a node is in proximity to a chain of connected edges (b), it can interact with multiple edges, or just an end point. The process for resolving rigid body motion used in the models. Any force applied to the edge (i) can be replaced with a force at the center of drag and a moment (ii). This is then used to calculate the ghost rigid body displacement that an isolated edge would experience in a given time step (gray edge in (iii)). From this, we reverse engineer the equivalent forces that would produce the same resultant displacement of the nodes when they are treated independently of the edge (iv).

Interactions act along the vector pointing from the node to the contact point, whereas the force magnitude depends on the length of this vector. Equal and opposite forces are applied to the node and the contact point. In a node–node interaction, the resultant forces are applied directly to the nodes; however, if the contact point is an interior point we must invoke rigid body mechanics. As we are assuming inertia is negligible compared with viscosity, we need to use a form of rigid body mechanics constructed around viscous drag coefficients. This is not something readily found in relevant texts, hence for the purpose of this work it has been developed from first principles. We introduce the important concepts here and provide further details in Methods. We also provide relevant derivations in Supplementary Section 1.

Figure 1c illustrates how rigid body motion is realized. Any force applied to a rigid body (Fig. 1c(i)) can be replaced with a force applied at the center of drag (analogous to the center of mass), and a moment (Fig. 1c(ii)). We use the viscous rigid body equations of motion to determine the ghost displacement of the edge in a given time step (Fig. 1c(iii)), by treating it as an isolated edge. We then reverse engineer two equivalent forces that would result in the same ghost displacement through purely linear motion of the endpoints (Fig. 1c(iv)). The equivalent forces due to edges—as well as any direct or internal forces—are then summed for each node and the simulation is evolved in time using the usual equation of motion (equation (i)).

To illustrate the capabilities of our rigid body framework, we present four application exemplars: a polygonal cell model used to simulate a tumor spheroid, an augmented vertex model applied to epithelial buckling where the layer can self-interact, a model using isolated edges to represent bacteria, and a membrane model used to investigate a tumor confined to a duct.

Exemplar 1 (tumor spheroid model). Using the rigid body framework, we can build a cell model that combines the the clear definition of cell area and shape of the vertex model with the freedom of movement seen in the overlapping spheres model. We construct a cell using a collection edges to form its boundary, but rather than sharing these with adjacent cells (as in the vertex model), each cell has its own set. Intercellular forces use the node–edge interaction mechanism, allowing smooth reactions and deformations, while also permitting total separation of cells. A suitable pedagogical setup to test this model is a tumor spheroid.

Here we model the growth of a tumor spheroid using decagonal cells built from edges. Interactions between cells are governed by the node–edge interaction mechanism (Fig. 2f), whereas internal forces are represented using energy methods akin to those used by Nagai and Honda. Cell growth follows a basic cell cycle model and division is modeled by cutting the boundary into two halves and turning the halves into new cells by adding edges (see Fig. 2e). We also implement contact inhibition to prevent cells from commencing growth when they are under high compressive stresses. Full details are found in Methods. We run 20 simulations from random seeds to calculate average values.

Figure 2a shows the time progression of a spheroid using this decagonal cell model. The snapshots show that as the tumor grows in volume, the interior cells become too compressed to start growing, whereas the cells in a band around the perimeter are able to grow and proliferate.

Figure 2b shows the growth in cell number, N, whereas Fig. 2c plots the radius capturing 90% of the cell centers, both averaged over 20 simulations. After the cell population has reached a reasonably circular shape (roughly $t > 100h$), the radius grows at an approximately constant rate, implying the cell population is increasing quadratically. Fig. 2d shows the average cell area as a function of distance from the spheroid center. As time progresses, the inner compressed core expands, while proliferation is limited to a narrow band around the rim. These population-level behaviors agree with other models and experiments. We also see some features here that may not be possible in other models. The inset for $t = 160h$ of Fig. 2a shows a spontaneously forming a rosette, an important feature in epithelial tissues where four or more cells form a junction. In a vertex model, rosette appearance needs to be specifically included in the model by modifying the way node swapping is handled, whereas here it has formed as a natural consequence of the interaction mechanism. More generally, we see that cells are able to take on a range of shapes depending on their local environment. Cells nearer to the proliferating rim are less compressed and hence closer to a regular decagon in shape, whereas cells in the central region have more varied shapes, some with an elevated aspect ratio, others approaching rectangles or pentagons.

Exemplar 2 (epithelial monolayer model). One of the most useful capabilities of the rigid body framework is its ability to allow tissues to self-interact without the need for interaction procedures that limit the scale and realism of the model. As an example, we look at a deforming epithelial monolayer.
Here we implement an epithelial monolayer vertex model similar to that used by Merzouki and colleagues\(^1\). Rectangular cells are joined in a contiguous ring, and follow the basic cell cycle model described in Methods to control their growth and division (Fig. 3c) with the same parameters as the polygon cell model; however, unlike previous work, we use the node–edge interaction mechanism to model interactions when the monolayer self-contacts (Fig. 3d), by directly modeling the boundary with edges.

Figure 3a shows the progression of this model as it grows and buckles. Early times show the usual behavior where the layer buckles produce finger-like extrusions. At around \(t = 60\) h we start to see the first stages of self-contact, where the layer is interacting with itself, but the deformation is small. By \(t = 75\) h (Fig. 3a) the node–edge interaction mechanism has allowed the layer to become extremely warped and still progress in a mechanically consistent way. The interactions are able to dynamically reconfigure themselves as nodes slide past, without causing them (or the edges) to become locked in place, all while allowing movement energy to be transferred across the contacts.

To track the state of the simulation we calculate the circularity of the monolayer

\[
C = \frac{4\pi A}{P^2},
\]

where \(0 \leq C \leq 1\) and \(C = 1\) for a circle; \(A\) is the internal area of the ring and \(P\) its internal perimeter. We see that the node–edge interaction mechanism allows the simulation to progress to the point where circularity approaches zero (Fig. 3b).

Other models (for instance, ref. \(^1\)) have relied on identifying collisions and resolving them using a procedure to handle self-contact. This can be computationally intensive if implemented naïvely,
although efficient algorithms do exist\textsuperscript{19}; however, these procedures do not account for force transfer between components, limiting the realism of the model. By modeling cell boundaries with edges and introducing the node–edge interaction mechanism, forces can be transferred between non-joined cells using physical principles, leading to more realistic motion of contacting tissues. Furthermore, as the node–edge interaction mechanism functions regardless of cell connectivity, the epithelial layer can interact with other objects in the environment such as supporting structures. This opens up the possibility of modeling how epithelial layers interact with their underlying stromal tissue to produce dynamically stable structures, something that has been identified as an underdeveloped area in cell-based modeling\textsuperscript{20}.

**Exemplar 3 (bacterial cell model).** At the simplest level, we can use the rigid body framework to produce an overlapping rods model, using isolated edges to represent cells such as bacteria and yeast. Using this model we are able to reproduce several features observed by Volfson and colleagues, in which microbes proliferate in a monolayer, constrained to a channel (Fig. 4; see Methods for details)\textsuperscript{21}. Figure 4a shows several snapshots of our model. Initially, cells are randomly placed in the channel in a random orientation. As the population grows, the cells increasingly make contact with the boundaries, causing them to align with the channel axis. This alignment propagates into the channel interior, particularly at the proliferating front.

In Volfson et al. the authors use the order parameter

$$Q = \sqrt{\sum_i \cos(2\phi_i)^2 + \sum_i \sin(2\phi_i)^2}$$  \hspace{1cm} (3)

borrowed from liquid crystal theory\textsuperscript{22} to measure the orderliness of the cell population. Here \(\phi_i\) is the angle cell \(i\) makes with the horizontal. This captures whether the cells are oriented in the same direction \((Q \approx 1)\), or have a random orientation \((Q \approx 0)\). Figure 4b shows how \(Q\) changes with time, averaged over 20 simulations, each starting with a population of 20 randomly oriented cells. During
the initial stages, the orderliness varies quite widely, but on average is $Q \approx 0.3$. From approximately $t = 60$ min, the orienting influence of the walls starts to propagate, leading to a higher overall orderliness. This qualitatively agrees with the findings by Volfson and colleagues. The authors also note that as time progresses, the average size of the cells decreases. Fig. 4c shows that this behavior is replicated in our model. This is probably a result of contact inhibition preventing growth from starting, and increased compression due to the densely packed cells.

The behavior we see here is not unique to the overlapping rods model, it can also be produced in CellModeller\cite{23}; however, this exemplar demonstrates that we are able to reproduce established results in a more versatile framework that supports other modeling applications beyond bacterial cells.

Exemplar 4 (membrane tissue model). Finally, we demonstrate how the rigid body framework can be used to model membranes. In 2010, Norton et al.\cite{24} used a two-dimensional overlapping spheres model to investigate the formation of ductal carcinoma in situ. To provide a supporting structure, they model the surrounding myoepithelial cells with several layers of nodes. As an example, we replicate this model, except in place of the myoepithelial cells we model the basement membrane using conjoined edges. Forces between cells follow the simple overlapping spheres approach\cite{12}, whereas forces between cells and the membrane use the node–edge interaction mechanism (Fig. 5f). Forces are applied to the edges to provide tension in the membrane, pushing it towards a resting circumference. Cell growth is handled by controlling the separation between two cells in the growth phase, and division occurs by releasing this control when growth is complete (Fig. 5d). We apply internal pressure in the lumen according to the procedure outlined in Methods. We run 20 simulations for a range of membrane tensions and average the results.

Figure 5a shows the cross-sectional model of the duct as the cells proliferate into the lumen, with the black line illustrating the...
basement membrane. The internal space gradually fills until around $t = 65$ h when the lumen closes up. Figure 5b shows the lumen area as a function of time for varying values of the membrane edge tension, $s_e$. Weaker membranes show a more pronounced increase in lumen area initially, but all eventually tend to zero. Notably, we see that lumen closure occurs at the same time on average for each value of $s_e$ shown. The total contained area of the duct (Fig. 5c) stays relatively constant while the lumen is open, except for a brief increase initially when the membrane tension and internal pressure equilibrate. The membrane is resisting the (relatively small) forces due to proliferation. After approximately $t = 65$ h when the lumen closes, the total area of the duct increases to accommodate the growing cell population. Eventually proliferation completely stops due to contact inhibition and the compressive forces from the membrane tension. Here we can see that a weaker membrane tension results in a larger tumor size before proliferation ceases.

Norton et al. model a supporting structure by using multiple layers of overlapping spheres that have a radius half that of the spheres used to represent the normal epithelial cells. They state the multiple layers are necessary to prevent the epithelial cells breaking through the supporting structure. The smaller cell size in the structure presumably serves a similar purpose. Images of late-stage
simulations seem to show that this layer has stretched to single cell in thickness, which could arguably result in the layer being breached in subsequent time steps. By contrast, we are able to produce the same effects with our membrane model made of a single layer of edges. Large internal pressures can be contained without concern for the structure breaking down, even with small membrane tension forces.

Discussion

In this work we have presented a modeling framework that expands the possibilities for cell-based modeling by introducing a versatile way to model edges in a system dominated by viscous drag forces. It can be seen as the natural progression from popular node-based models in which the foundational object is now a one-dimensional edge rather than a node. This is important as edges offer a much more intuitive way to model objects with length than a node-based framework.

A key capability of the framework—setting it apart from existing models/tools—is that we are able to join edges together to produce contiguous surfaces, as seen in Exemplars 1, 2 and 4. Some existing models construct surfaces out of nodes (for example, ref. 23). In this situation, there is fundamentally nothing stopping one node from moving between any pair of adjacent nodes. This is a major limitation to modeling barriers as their function is to separate one region of space from another. If a node approaches a barrier built from edges, the barrier can react as if a force is applied directly to its interior, rather than approximating the interaction directly with the nodes. Other approaches to modeling barriers have used front-tracking techniques such as the level set method24, which mathematically represents a boundary; however, these techniques can be computationally intensive25, limiting them to small numbers of cells. Edges can cover suitably large regions of space and still provide a robust barrier response, and are computationally efficient enough to handle hundreds or thousands of cells.

Robust edge-based barriers have several useful applications, as demonstrated in this work. The epithelial ring model (Exemplar 2) demonstrates that a vertex model using connected edges to model the cell boundaries can allow distant parts of a tissue to self contact without the complicated swapping procedures of the standard vertex model23. We can exploit this ability to allow separately defined vertex models (with different internal mechanisms) to interact. This can be used to model deformable tissues in close contact that have different properties such as stroma and epithelia in the gut26. Deformability of contacting tissues has been identified as an underdeveloped area for multicellular modeling that limits its ability to model certain tissues27. The rigid body framework presents a tool that addresses this gap. This has important applications for modeling morphogenesis (for example, neurulation28) and for modeling tissues where an epithelial layer may be tightly folded (such as the intestinal crypt29).

Another benefit of the rigid body framework is that we can model cells with arbitrary shapes without node sharing, generating tessellations or internal nodes. In a vertex model, cell shape is defined clearly, however, the need to share nodes between multiple cells requires special processes for cells to move through a tissue30. Tessellation models30 can produce a clear shape for a cell, but this depends on connections with adjacent cells, which complicates matters for sparse cell populations and tissue boundaries. The subcellular element method31 uses nodes to fill its interior to provide material properties, increasing the computational load per cell. It also does not enable torques to be applied to connected elements. With the polygon cell model (Exemplar 1), cells can easily move through tissues, without needing to consider how nodes are shared. Cells can also maintain their shape, regardless of connectivity networks, and they can do this efficiently by only modeling the surface. Rather than reacting as a line of connected springs, as in the SEM32, the surface locally rotates due to applied torques. Polygon cell models can also become completely isolated from other cells while maintaining their shape. All interactions are determined by neighborhood searching, meaning surface interaction is built into the polygon cell model. The ability to use neighborhood searching also has benefits in terms of computational time. Efficient and relatively simple algorithms exist for finding nearest neighbors that scale linearly with number of nodes and number of neighbors (that is, $O(kn)$ time)33. It can be shown (Supplementary Section 2) that edges do not add any extra computational burden to neighbor searching.

As with all multicellular modeling, the rigid body framework is not without its own drawbacks. The overlapping rods model suffers from the same poor definition of shape as the overlapping spheres model. It also has the risk of rods crossing, creating a nonphysical situation, although this is mitigated by choice of force law. For polygonal cells and membranes, special attention is needed to ensure the boundary remains a simple polygon, where it does not cross itself. This can be achieved by implementing additional forces to oppose such motion.

There are further potential applications for the specific models presented here. The membrane model can be used to model any kind of thin structure that separates regions, whether it forms a ring or a flat surface. The polygon cell model (which can be considered a special case of a ring membrane) can be adapted to model any large volume of tissue with internal properties. Details on specific potential projects can be found in Supplementary Section 2.

Further development of our framework and associated tool will aim to integrate common multicellular modeling features such as diffusible species and subcellular dynamics, and extend it to three-dimensional modeling. In the future we plan to integrate our framework into the open source library, Chaste33.

Perhaps the most promising future application is in three-dimensional multicellular modeling. Large-scale three-dimensional modeling, with many thousands of cells, is well served by the overlapping spheres model34. Smaller-scale modeling, in which the structure of a single cell is the focus, is well served by subcellular and finite element methods35. The intermediate scale, where cell shape can have important consequences, but where the number of cells makes the more detailed approaches intractable, is currently underdeveloped and a perfect fit for our framework.

Methods

Application details. Exemplar 1 (tumor spheroid model). In the wet lab, tumor spheroids are used to investigate the early behavior of cancer36. A cell line is allowed to grow in vitro, creating a cluster of cells that consume nutrients and oxygen as they proliferate. A common observation is the halting of proliferation in the core of the spheroid as its size increases, which is understood to be caused by dwindling resources and increasingly dense packing of the cells37. Spheroids are among the simplest biological systems to translate into a computer model. They have been examined in many works38, from a mechanical investigation of growth with oxygen diffusion39, to detailed models that include vascularization40.

Using the rigid body framework, we investigate a tumor spheroid by using an arbitrary decagon to represent a cell. The boundary of the cell is made up of a contiguous ring of edges; hence, each edge shares its endpoints with two others. Interactions between cells (Fig. 2f) are determined using the node–edge interaction mechanism. For computational efficiency, direct interactions between nodes are not considered and this does not affect the results as discussed in the ‘Practical model implementation details’ section. Forces between nodes and edges are calculated using a standard force law (equation (21)), and for illustrative purposes we allow cells to overlap by setting $d_{	ext{area}} < 0$. Cells are driven to a target shape using the the energy methods from equation (29). The target cell area is determined by the cell cycle model, and the target perimeter is driven by the target area by assuming the cell wants to form a regular decagon. The parameters used are collected in Supplementary Table 1.

In this model a cell maintains a constant number of edges around its boundary during its life cycle. It is possible to make the number of edges depend on cell size for the sake of computational speed or shape accuracy; however, this requires special attention to how cell boundaries are modified, and is left for future work.

With the standard growth-focused cell cycle model32, the target area of the cell. A new cell starts off with target size $S_0$ and remains at this target size until $t_c$, which is chosen from a uniform distribution such that $t_c \in [8, 12]$.h.
Contact inhibition holds the cell in this resting state until the size of the cell, $s$, satisfies $s \geq s_0$, where, in this case, $\gamma = 0.9$. Growth occurs over a period of time $t \in [U, 12] \text{ h}$, during which the target area increases linearly from $S_0$ to $S_{\text{target}}$, allowing it to be approximated as the area of an ellipse. We divide the growth period into three distinct phases: 1) a spherical phase, in which two daughter cells occupying the same space as the parent cell. We choose a cell diameter (CD) of 15 $\mu$m to roughly match the size of a standard HeLa cell. New cells will have a target size of $S_t = 0.5 \text{ CD}^2$ and fully grown cells will be $S_t = 1 \text{ CD}^2$. The total cell cycle duration ($t + \gamma$) is in the range of that reported for HeLa cells of about 22 h (ref. 21).

For a polygonal cell, the process of creating new cells through division has several steps (Fig. 2e). A random vertex is chosen to define the division axis, and the cell is split in half so that each daughter cell takes five of the edges from the parent cell. Five new edges of equal length are added in a straight line between the loose ends of each half to close the loop and form the new cells. The newly divided daughter cells will be slightly shifted on the plane, and the division center will be a point on the plane with a random offset in area and a perimeter somewhat longer than the target perimeter due to the offsetting. This causes the new cells to reconfigure themselves over several subsequent time steps into a form that more closely resembles the physical shapes observed in vitro post division.

By using this simplified method of cell division, the distribution of edges around the boundary of the cell can have a significant impact on the shapes of the newly divided cells. In a worst-case scenario, this can leave one cell with most of the parent cell’s area, and the other may be just a small sliver, potentially causing the new cells to reconfigure themselves over several subsequent time steps into a form that more closely resembles the physical shapes observed in vitro post division.

By using this simplified method of cell division, the distribution of edges around the boundary of the cell can have a significant impact on the shapes of the newly divided cells. In a worst-case scenario, this can leave one cell with most of the parent cell’s area, and the other may be just a small sliver, potentially causing the new cells to reconfigure themselves over several subsequent time steps into a form that more closely resembles the physical shapes observed in vitro post division.

Exemplar 2 (epithelial monolayer model). Epithelial monolayers are a common tissue found in the human body and throughout the animal kingdom (ref. 38). Due to their simplicity, monolayers have found much attraction from cell-based modelers, with common questions relating to the buckling behavior of the layer due to cell growth. Many models allow an epithelial layer to grow and buckle (refs. 39–43), but the buckling can usually only continue as long as no self-contact occurs, particularly when vertex models are used. Very few models have been able to produce simulations that can meaningfully continue for long-term, large-scale self-contacting deformations. This has meant that modeling morphogenesis beyond self-contact has been difficult to achieve. Merzouki et al. (ref. 18) use a phenomenological procedure in their epithelial vertex model to handle self-contact. When a node (vertex) moves into an adjacent cell, the non-physical situation is resolved by moving the node back to the edge that it crossed. Although this prevents unrealistic cell overlapping, the algorithmic response means that forces are not transferred between interacting cells. Such behavior causes energy to be lost from the simulation that might otherwise have produced appreciable movements. A mechanical resolution to the interactions is therefore more desirable.

To examine a buckling epithelial monolayer, we integrate a simple vertex model similar to that used by Merzouki et al. into the rigid body framework. There, parameters from Supplementary Table 1 are applied, and the simulation is allowed to run for 200 h. We run 20 simulations, each starting from a unique random seed, and use them to calculate average population behaviors.

Exemplar 3 (bacterial cell model). Here we treat each edge as a separate agent where end-points are not shared with other edges. In doing this, we can reproduce the modeling capabilities due to Budgie et al. in CellModeller (ref. 1), specialized software tools used for modeling complex cell division and growth. The model in three dimensions uses similar underlying equations, but drives the simulation by energy minimization with an iterative solution to a matrix equation. To demonstrate overlapping rods, we examine orientation of bacterial cells.

In 2008, Volfson et al. (ref. 21) investigated cell behavior in a confined environment by constraining E. coli cells to a monolayer in a microchannel. They were able to demonstrate that the spatial constraints of the environment influenced the orientation of the cells; specifically, cells tended to align with the axis of the channel. Here we replicate the experiment using an overlapping rods model based on the viscous equations of motion. Cells are modeled as rods with a basic cell cycle that controls their growth. Internal forces drive the cell to target length via a force applied to the rod. Division occurs when the cell reaches a fully grown cell covering the same space (Fig. 4d) and interactions are determined via neighborhood searching (Fig. 4e). Contact inhibition is implemented to prevent growth commencing when a cell is over compressed. Full details are given below. We run 20 simulations each with a random seed and calculate average values.

When a cell in the monolayer has satisfied the conditions for division (the same as in the spheroid model, these interactions are repulsion only, so we set the attraction parameter $a = 0$. According to Volfson et al. an average cell length is plotted in Fig. 4c, demonstrating this choice achieves the required aspect ratio range.

We apply a standardized cell cycle model designed to control the target size of the cell (see the ‘Cell growth and division’ section below). This is different to those used in other works for rod-shaped cells, but we use it here for consistency with the other applications. A new cell starts off in a resting state in which the target rod length is constant at $S_t$. It remains in this resting state until it reaches the age $t$, which is chosen from the uniform distribution $t \in [0, 2] \text{ min}$. If the measured size of the edge at is at least $s \geq S_t$, where $\gamma = 0.9$; the cell is then allowed to start growing, otherwise it waits in a contact inhibited state until the size requirement is met. Growth is modeled by linearly increasing the target size from $S_t$ to $S_{\text{grown}}$ over a period of time $t_{\text{g}}$, which is also chosen from a uniform distribution so that $t_{\text{g}} \sim [U, 12] \text{ min}$. Division occurs immediately after $t_{\text{g}}$. The total cell cycle duration ($t + \gamma$) was chosen so that the overall population behavior (accounting for contact inhibition) had a similar turnover rate compared to that reported in Volfson et al., which produced about three generations in 60 min.

Division produces two collinear, equal-length daughter cells that fill the same space in the parent cell. This is achieved by splitting the edges with a small gap separating them at the preferred separation distance (Fig. 4f).

Division produces two collinear, equal-length daughter cells that fill the same space in the parent cell. This is achieved by splitting the edges with a small gap separating them at the preferred separation distance (Fig. 4f). Cells arranged like this may remain perfectly collinear in the absence of outside forces, but we prevent this with addition of random noise when the motion is calculated (see the ‘Practical model implementation details’ section). After division, the cell cycle for both cells starts from $t = 0$, hence the new target size will be $S_t$. Division produces two collinear, equal-length daughter cells that fill the same space in the parent cell. This is achieved by splitting the edges with a small gap separating them at the preferred separation distance (Fig. 4f).

Exemplar 4 (membrane tissue model). Membranes are seen in numerous tissues (ref. 44), including the colonic crypt (ref. 45) and the breast ducts in mammals (ref. 46). The membrane is often approximated as a fixed supporting structure that does not move under load (for example, ref. 47); however, the dynamic response of a membrane can be important to the behavior of the tissue. Previous work has exploited collections of nodes to represent the membrane. Nodes can be used indirectly to determine a force that produces membrane-like behavior as, for instance, in Dunn and co-workers (ref. 48). More often, they are used to directly represent a membrane. A notable example is Buske and colleagues (ref. 49), who model a three-dimensional organoid with a network of nodes representing a thin supporting structure. In the direct case,
interactions with the membrane evidently need to be applied to the nodes. This will not necessarily be the closest point on the membrane (Fig. 5c), which can have consequences when the membrane is used to apply repulsive forces. Degenerate cases can occur where the angles between the edges and cell nodes mean it is all but impossible for the node–node forces to push cells away. This can result in cells under load breaching the barrier. We can avoid this issue entirely by using rigid edges to represent the membrane, where forces can be applied between the node and the closest point on the edge.

To investigate the growth of a tumor in a membrane, we integrate an overlapping spheres model into the rigid body framework. Each cell is represented by a single node with a preferred separation from other nodes or edges. We use the node–node interaction mechanism for cell–cell forces to produce the same behavior as seen in the simple overlapping spheres model. This can also be considered a limiting case for interactions with isolated edges, where the length of the edge tends to zero (see the ‘Implementation details: force magnitude’ section). The preferred separation between cell centers (that is, the nodes) is set to $d_{sep} = 1$ CD, where CD = 10 μm which is in line with cells found in the duct. This allows the boundary of the cell to touch at a single point when they are under no forces. When a cell starts to grow, a second node is introduced a small distance from the original node, and the preferred separation between these nodes is gradually increased up to $d_{sep} = 1$ CD over the course of the growth phase $t_g \approx [7, 11]$ h (Fig. 5d). Division occurs at the end of the growth phase, and it is realized by removing the special control over $d_{sep}$ between the two nodes. After division, the new cells stay in the non-growing pause phase for $t_g \approx [7, 11]$ h, where $d_{sep}$ remains constant. Contact inhibition stops the cell from commencing growth unless its calculated size is large enough, that is, $s \geq s_y$. The size of a node cell is calculated using the process outlined in the ‘Implementation details’ section, where the proximity of adjacent cells and edges is considered. As growth is handled by controlling $d_{sep}$, we do not need to consider area calculation for growing cells. The total cell cycle length is on average 18 h, which is in line with that used for similar simulations.

The membrane is represented as a contiguous ring of edges, in much the same way as the boundary of a polygonal cell. Interactions between the cells and the membrane are determined using the node–edge interaction mechanism and forces are calculated using equation (21). The preferred separation between cells and the membrane is set to $d_{sep} = 0.5$ CD. This means that in the absence of other forces, the boundary of the cell will contact the edge tangentially at a single point. The number of edges in the membrane is fixed at 40 throughout the whole simulation, making a circular ring that we fix at a point. The number of edges in the membrane is fixed at $40$, throughout the whole simulation, making a circular ring that we fix at a single point. When a cell starts to grow, a second node is introduced a small distance from the original node, and the preferred separation between these nodes is gradually increased up to $d_{sep} = 1$ CD over the course of the growth phase $t_g \approx [7, 11]$ h (Fig. 5d). Division occurs at the end of the growth phase, and it is realized by removing the special control over $d_{sep}$ between the two nodes. After division, the new cells stay in the non-growing pause phase for $t_g \approx [7, 11]$ h, where $d_{sep}$ remains constant. Contact inhibition stops the cell from commencing growth unless its calculated size is large enough, that is, $s \geq s_y$. The size of a node cell is calculated using the process outlined in the ‘Implementation details’ section, where the proximity of adjacent cells and edges is considered. As growth is handled by controlling $d_{sep}$, we do not need to consider area calculation for growing cells. The total cell cycle length is on average 18 h, which is in line with that used for similar simulations.

Parameter selection. Supplementary Table 1 shows a summary of all the parameters used in the models presented here. Where available, they have been chosen based on physical values reported in literature. This has been limited to values related to cell size, and cell cycle duration as referenced in the ‘Application details’ section above. However, many of the parameters do not have direct biological analogs. In particular, the parameters related to force calculation are difficult to tie to specific published biological data. They have therefore been chosen so they produce stable simulations with qualitative behavior that resembles the biological system of interest. The simulations are largely robust to variations in mechanical parameters such as spring stiffness and energy parameters (plus or minus an order of magnitude), although large values can require changes in numerical parameters (smaller time step sizes) to remain stable.

Implementation details. Viscous rigid body motion. At the core of this framework, we are assuming that the system is over-damped, where drag forces dominate. Surprisingly, equations of motion for overdamped rigid body mechanics are (4) and (5) for the moment of inertia; however, these concepts are ill-defined when inertia is neglected. A rigid body theory balancing applied forces with drag needs equivalent concepts that use drag coefficients instead of mass.

The first of these concepts, we define as the total drag, $\eta D$. This is the drag, $\eta D$, integrator over the body

$$\eta D = \int_{R} d\eta, \hspace{1cm} (4)$$

where $R$ defines the body region and $d\eta$ is the infinitesimal unit of drag. We also define the vector $r$, that tracks the center of drag, $D$, by

$$r = \frac{1}{\eta D} \int_{R} r d\eta, \hspace{1cm} (5)$$

As with the center of mass, the center of drag is the point where we can consider the total drag of a rigid body (or system of particles) to be concentrated. If we apply a force that passes through the center of drag of a rigid body, it is possible to show that there are no resulting rotations (Supplementary Section 1), permitting us to treat the body as a point object. This allows us to use the linear motion of a rigid body in an over damped system by the equation

$$\frac{dF}{dt} = \sum F_j, \hspace{1cm} (6)$$

where $F_j$ are the external forces applied to the body at $D$.

If a force does not pass through the center of drag, then the body will also experience a rotation. When inertia is dominant, the rotation is resisted by the angular momentum, however, when drag forces dominate, we use the quantity angular drag, $H_D$, to resist motion, which in two dimensions is defined by the scalar

$$H_D = \frac{d\eta D}{dt}. \hspace{1cm} (7)$$

where $\theta$ is the angular position of the body relative to some axis, and $I_\theta$ is its moment of drag about $D$. This can also be calculated in three dimensions, however, we need to consider moments about all three axes, resulting in a vector form of $H_D$. We leave this for future work. Continuing in two dimensions, we define $I_\theta$ by

$$I_\theta = \int_{R} r \cdot r d\eta, \hspace{1cm} (8)$$

where $r$ is the distance from $D$ to $d\eta$. The angular drag balances with the externally applied moments, $M_\theta$, giving us the second equation of motion

$$\frac{d\theta}{d\tau} = \sum M_\theta, \hspace{1cm} (9)$$

where $\theta$ is a unit vector perpendicular to the plane of the cells. This allows us to determine the rotation of the body.

For a rigid body with a single externally applied force $F$ that does not pass through the center of drag, it can be shown that $\theta$ can be replaced by a force acting at the center of drag and a moment $\theta$. The motion of the body can thus be completely determined by evaluating the two equations

Motion of an edge. Supplementary Fig. 1 shows an isolated edge in an overdamped system subject to an external force. Vector $r$, tracks the position of the center of drag, and the vectors $r_B$ and $r_C$ track the positions of the rod’s endpoints relative to $D$. We can then construct vectors that track the position of the endpoints of the edge, $r_1 = r_1 + r_B$ and $r_2 = r_2 + r_C$. We only consider forces that act perpendicular to the edge, hence we introduce the force $F_\perp$, which is applied to the edge at $A$, the point of action. To solve the motion of the edge, we need to carefully apply equations (10) and (11).

Given the vector quantities described above, we can expand equations (10) and (11) with the relevant scalar components. For the linear motion, we write the scalar components relative to the fixed system of coordinates $(X, Y)$, and for the rotation it is convenient to write the components relative to the rotating system of coordinates $(x, y)$, as we are interested in the position of the endpoints relative to $D$. Noting that $r_{DA, X} = 0$, these end up being

$$\frac{dF_{DA, X}}{d\tau} = \frac{F_{DA, X}}{\eta D}, \hspace{1cm} (10)$$

and

$$\frac{dF_{DA, Y}}{d\tau} = \frac{F_{DA, Y}}{\eta D}, \hspace{1cm} (11)$$

where the character following the comma in the subscripts represents the component in the respective coordinate system. The total drag, center of drag, and moment of drag are calculated depending on how the edge is modeled. In the examples in this paper, we assume the drag is concentrated at the endpoints, hence

$$\eta D = \eta_1 + \eta_2, \hspace{1cm} r_0 = \frac{1}{2} r_0 \eta_1 + \eta_2, \hspace{1cm} \text{and} \hspace{1cm} I_\theta = \eta_1 r_{DA, X}^2 + \eta_2 r_{DA, Y}^2. \hspace{1cm} (12)$$
To actually solve these for the subsequent motion, we need to apply an intermediate step to deal with the rotation.

Converting moments into forces. As is commonly used when solving motion in cell-based models, we apply the forward Euler method to determine the rotation and position at the next time step of a simulation. The motion of the center of drag is trivially solved at the next time step by

\[
\mathbf{r}_\text{drag} \left( t + \Delta t \right) = \mathbf{r}_\text{drag} \left( t \right) + \frac{\Delta t}{\mathbf{J}} \mathbf{F}_\text{drag},
\]

(14)

We can also quickly find the change in angle of the edge using this method

\[
\Delta \theta = -\frac{\Delta t}{\mathbf{J}_\text{edge}} \text{D}_\text{edge}, \quad \text{D}_\text{edge} = \text{D}_\text{edge}(t) + \frac{\Delta t}{\mathbf{J}_\text{edge}} \text{D}_\text{edge}(t).
\]

(15)

but to find the new position of the edge, we must use the rotational transformation matrix defined by

\[
\mathbf{R}(\Delta \theta) = \begin{bmatrix} \cos(\Delta \theta) & -\sin(\Delta \theta) \\ \sin(\Delta \theta) & \cos(\Delta \theta) \end{bmatrix}.
\]

(16)

This rotates the body system of coordinates (that is, the axes \((x, y)\)) within the fixed coordinate system (that is, \((X, Y)\)). As the position of the edge is uniquely determined by its endpoints, the goal of the procedure is to find the positions of the endpoints \(j\) at the next time step. Using the rotation matrix, the position vectors of the endpoints relative to \(D\) become

\[
\mathbf{r}_j(t + \Delta t) = \mathbf{R}(\Delta \theta) \mathbf{r}_j(t), \quad j \in \{1, 2\}.
\]

(17)

The final position of an endpoint due to the force applied to the edge is found by superimposing the two movements. We are able to superimpose these movements because the linear movement represents the new position of the center of drag, whereas the rotational movement expresses how the body is oriented. The motion can be split up this way because a force applied at a distance from the center of drag can be represented as a force at the center of drag, plus a moment.

\[
\mathbf{r}_\text{edge} \left( t + \Delta t \right) = \mathbf{r}_\text{edge} \left( t + \Delta t \right) + \mathbf{r}_\text{drag}(t + \Delta t).
\]

(18)

For an edge with a single applied force, this is sufficient to solve its motion, so \(\mathbf{r}_\text{edge} \left( t + \Delta t \right) = \mathbf{r}_\text{edge} \left( t + \Delta t \right)\), resulting in true rigid body behavior; however, if the edge is connected to one or more other edges (as would be seen when modeling cell boundaries), rotation cannot be applied so easily. By moving the endpoint of one edge, we will necessarily be altering the state of any other edge sharing that endpoint. To accumulate the motion due to each edge, we define the rod force

\[
\mathbf{F}_\text{edge} \cdot x = \frac{\eta}{2} \left( \mathbf{r}_\text{edge} \cdot x \left( t + \Delta t \right) - \mathbf{r}_\text{edge} \cdot x(t) \right).
\]

(19)

This is the equivalent force that would push the endpoint \(j\) of edge \(k\) to the position \(\mathbf{r}_\text{edge} \cdot x \left( t + \Delta t \right)\) through purely linear motion. To collect the forces on an endpoint, we need to consider global indices that identify the given node and edge in the context of the whole simulation. If \(l\) represents any node, and \(k\) identifies each edge that has \(l\) as an endpoint, then we define the index function \(j(l, k)\), which identifies the position of \(j\) in \((k\) (that is, whether it is endpoint 1 or 2). The total equivalent force on each endpoint, due to the rotation of the edges is then

\[
\mathbf{F}_J(\text{Rotation}) = \sum_k \mathbf{F}_\text{edge} \cdot x(l, k).
\]

(20)

In effect, we have converted the rotation of an edge into a force that treats each endpoint as an individual point object. Under this definition, the edge can no longer be considered a true rigid body, as, depending on the forces applied, the edge can expand and contract in length. To describe this situation, we introduce the term semi-rigid body: the forces applied to the body come from treating it as rigid, but the resultant motion need not conserve the length of the edge. As a consequence, special care is needed during a simulation to ensure any given edge maintains a sensible length. Furthermore, this makes integrating the whole system through time a simple task. We can use the same processes applied to node-based models as we have no requirement to directly handle rotations.

Finding interactions. A node–edge interaction occurs when the endpoint of one edge (a node) comes into proximity with another edge. In this situation, we can identify the closest part of the edge by taking a circle centered on the node, and expanding it until it first contacts the edge in question (Fig. 1a,b). This point, which we call the point of action, will be used as the point where any interaction forces are applied. The node and the point of action taken together form the line of action for any subsequent forces. If the point of action is on the interior of the edge, the line of action will be perpendicular to the edge, and if the point of action is an endpoint, then the line of action extends radially out from the endpoint (Fig. 1a).

When the edges are connected and form an external convex feature as in the top of Fig. 1b, the node will only interact with the endpoint. If on the other hand, the connected edges form a concave feature as in the left of Fig. 1b, a node can interact with both edges, but will not interact with the endpoint. These situations can alternatively be viewed by drawing an interaction box around the edges as done in Supplementary Fig. 2. The box extends a distance \(d_{\text{sep}}\) which is the limiting distance for interactions. If a node is inside a box, it interacts with the interior of the edge, otherwise it may interact with an endpoint. Practically, this is how nodes are screened for interactions in the software tool. In Supplementary Fig. 2 the upper node is close to the two edges, but sits in a wedge where it can’t interact with the interiors, hence it interacts with the shared endpoint. The lower node is in two over lapping interaction boxes, hence it interacts with the interior of both edges and not the shared endpoint.

In either of these cases, two forces are applied along each line of action, one at the point of action, and the other at the node. They are equal in magnitude and opposite in direction.

Force magnitude. The magnitude of the forces is determined by the position of the node within the region of interaction around the edge. Supplementary Fig. 3 illustrates how the regions of interactions are formed around an isolated edge (Supplementary Fig. 3a), and an edge that forms part of a cell boundary (Supplementary Fig. 3b). The dashed line indicates a preferred separation locus. It is found at a constant distance from the edge, \(d_{\text{sep}}\). If a node is located on the locus, no force is applied. In effect, this defines the size and shape of the cell. Inside the dashed line is the repulsion region, where a force pushes the node and the edge apart. Between the dashed and dotted lines is the attraction region, where a force will pull a node and edge together. The dotted line indicates the limit of interactions, and is found at a constant distance \(d_{\text{sep}}\) from the edge. Outside of this limit, no interactions are calculated. The distance \(d_{\text{sep}}\) represents the point where the force should asymptote to infinite repulsion. For an isolated edge (Supplementary Fig. 3c), we must have \(d_{\text{sep}} \geq 0\). This prevents isolated edges from overlapping, a situation which has no physical meaning. From a computational perspective, this is equally important for preventing a node getting pushed away on the wrong side of the edge as the direction of the force is dependent on which side the node approaches from.

For an edge that is part of a cell boundary (Supplementary Fig. 3d), preventing overlap is less important, as we have a clear way to orient the edge by using the inside of the cell. Depending on the specific force law, a node can be allowed to cross into the cell, or it can approach a hard barrier at the edge, hence we can choose \(d_{\text{sep}}\) to have negative values. In either case, there is only one preferred separation point on the line of action, found outside the cell, so the direction of the force does not depend on which side of the edge the node appears from.

A force law that encapsulates these requirements is

\[
F(x) = \begin{cases} 
 s_1 \log \left( \frac{d_{\text{sep}} - x}{d_{\text{sep}}} \right) , & x < d_{\text{sep}} \\ s_2 \left( \frac{d_{\text{sep}} - x}{d_{\text{sep}}} \right) \exp \left( \varepsilon (d_{\text{sep}} - x) \right) , & 0 < x < d_{\text{sep}}, \\ 0 , & x > d_{\text{sep}}.
\end{cases}
\]

(21)

where \(s_1\) and \(s_2\) are parameters setting the strength of the force in the repulsion and attraction regions, respectively, and \(\varepsilon\) controls the shape of the attraction force; in the models presented here \(\varepsilon = 5\). If \(s_1 = s_2\), then the derivative of \(F\) is smooth at \(d_{\text{sep}}\). Other force laws can be used that capture the required features.

Applying a force to rods or boundaries. The force \(\mathbf{F}_\text{I}\) was applied without specifying exactly where it came from. However, in the model we know that it comes from a node–edge interaction. To calculate the force, we need to consider the two possible cases, that is, the edge is isolated, or that it is part of a boundary.

If the edge is isolated and the force is due to being in proximity to node \(J\) (which is the endpoint of some other edge), then we will only consider \(|\mathbf{r}_J|\), the length of the vector from \(A\) to \(J\). This is due to the fact that we are not allowing isolated edges to cross. If we did allow edges to cross there is no simple way to determine how the forces should be applied to undo the crossing. We would need to track and record how each edge is oriented with respect to every other edge in its vicinity over multiple time steps. We will not attempt to do that here, so we only need to consider the magnitude of the vector separating the two points. The force is then determined by

\[
\mathbf{F}_J = -F(|\mathbf{r}_J|)|u| u,
\]

(22)

where \(F\) is the function defined in equation (21), and \(u\) is the unit vector from the point of action \(A\) to node \(J\), which is normal to the edge. The negative sign is necessary because we take repulsion force magnitudes to be positive.

When we consider an edge that is part of a boundary, we can allow edges to cross, but we must consider the interaction. The inside of the cell can be taken as a negative position with respect to the normal vector that points to the outside of the cell. To take advantage of this, we need to pay more attention to how we define the unit normal vector.
We can define a unit tangent vector $\mathbf{v}$ pointing along the length of the edge and, with this, we can construct a unit normal vector $\mathbf{u}$, by noting that for two vectors in the $(u,v)$ plane, $\mathbf{u} \times \mathbf{v} = \mathbf{e}_z$. This is a unit normal vector pointing upwards out of the plane when viewed from above. As by definition they are perpendicular, $\mathbf{u}$ and $\mathbf{v}$ are related by

$$\mathbf{u} = [v_y, -v_x].$$

As a result, we define the interior of the cell to be the domain to the left when traversing the boundary of the cell in an anti-clockwise direction. The force is then

$$\mathbf{F}_A = -F(\mathbf{r}_A) \cdot \mathbf{u} \mathbf{u},$$

where the dot product will evaluate to negative values when $J$ is inside the cell.

Applying a force to endpoints. As described above in the ‘Finding intersections’ section, the point of action is defined as the first point an expanding circle touches a given edge. Around an endpoint, this produces a curved region of interaction with a constant radius. Supplementary Fig. 4 shows the interaction region around an endpoint that is part of a cell boundary. Likewise, Supplementary Fig. 3a shows the curved region around the endpoints of an isolated edge. If a node falls into this part of the interaction region, the interaction force is applied directly to the endpoint. When this happens, we can calculate the force on $A$ in the same way as in equation (22):

$$\mathbf{F}_j = -F(\mathbf{r}_E) \cdot \mathbf{u} \mathbf{u},$$

as, by definition, in this case $A$ will lie on the endpoint.

The forces applied to a node are equal and opposite to the forces applied to an edge at the point of action; thus, the force applied to a node due to the edge is

$$\mathbf{F}_{ij} = -\mathbf{F}_{ji},$$

where $K$ is an edge that $i$ is not part of, and $A$ is the point of action on $K$ due to this node–edge interaction. Depending on the situation $\mathbf{F}_i$ may be defined by equation (22) or equation (24). This also applies to the opposite node in a node–node interaction calculated in equation (25).

In the models presented here, we resolve forces applied to a node without considering the impact to any edge it is part of. The nodes (endpoints) are treated as free objects in space, fitting in with the semi-rigid body concept arising from considering the impact to any edge it is part of. The nodes (endpoints) are treated as free objects in space, fitting in with the semi-rigid body concept arising from considering the impact to any edge it is part of. As a result, we define the interior of the cell to be the domain to the left when traversing the boundary of the cell in an anti-clockwise direction. The force is then

$$\mathbf{F}_A = -F(\mathbf{r}_A) \cdot \mathbf{u} \mathbf{u},$$

where the dot product will evaluate to negative values when $J$ is inside the cell.

As a result, we define the interior of the cell to be the domain to the left when traversing the boundary of the cell in an anti-clockwise direction. The force is then

$$\mathbf{F}_A = -F(\mathbf{r}_A) \cdot \mathbf{u} \mathbf{u},$$

where the dot product will evaluate to negative values when $J$ is inside the cell.

Applying a force to endpoints. As described above in the ‘Finding intersections’ section, the point of action is defined as the first point an expanding circle touches a given edge. Around an endpoint, this produces a curved region of interaction with a constant radius. Supplementary Fig. 4 shows the interaction region around an endpoint that is part of a cell boundary. Likewise, Supplementary Fig. 3a shows the curved region around the endpoints of an isolated edge. If a node falls into this part of the interaction region, the interaction force is applied directly to the endpoint. When this happens, we can calculate the force on $A$ in the same way as in equation (22):

$$\mathbf{F}_j = -F(\mathbf{r}_E) \cdot \mathbf{u} \mathbf{u},$$

as, by definition, in this case $A$ will lie on the endpoint.

The forces applied to a node are equal and opposite to the forces applied to an edge at the point of action; thus, the force applied to a node due to the edge is

$$\mathbf{F}_{ij} = -\mathbf{F}_{ji},$$

where $K$ is an edge that $i$ is not part of, and $A$ is the point of action on $K$ due to this node–edge interaction. Depending on the situation $\mathbf{F}_i$ may be defined by equation (22) or equation (24). This also applies to the opposite node in a node–node interaction calculated in equation (25).

In the models presented here, we resolve forces applied to a node without considering the impact to any edge it is part of. The nodes (endpoints) are treated as free objects in space, fitting in with the semi-rigid body concept arising from considering the impact to any edge it is part of. As a result, we define the interior of the cell to be the domain to the left when traversing the boundary of the cell in an anti-clockwise direction. The force is then

$$\mathbf{F}_A = -F(\mathbf{r}_A) \cdot \mathbf{u} \mathbf{u},$$

where the dot product will evaluate to negative values when $J$ is inside the cell.

Applying a force to endpoints. As described above in the ‘Finding intersections’ section, the point of action is defined as the first point an expanding circle touches a given edge. Around an endpoint, this produces a curved region of interaction with a constant radius. Supplementary Fig. 4 shows the interaction region around an endpoint that is part of a cell boundary. Likewise, Supplementary Fig. 3a shows the curved region around the endpoints of an isolated edge. If a node falls into this part of the interaction region, the interaction force is applied directly to the endpoint. When this happens, we can calculate the force on $A$ in the same way as in equation (22):

$$\mathbf{F}_j = -F(\mathbf{r}_E) \cdot \mathbf{u} \mathbf{u},$$

as, by definition, in this case $A$ will lie on the endpoint.

The forces applied to a node are equal and opposite to the forces applied to an edge at the point of action; thus, the force applied to a node due to the edge is

$$\mathbf{F}_{ij} = -\mathbf{F}_{ji},$$

where $K$ is an edge that $i$ is not part of, and $A$ is the point of action on $K$ due to this node–edge interaction. Depending on the situation $\mathbf{F}_i$ may be defined by equation (22) or equation (24). This also applies to the opposite node in a node–node interaction calculated in equation (25).

In the models presented here, we resolve forces applied to a node without considering the impact to any edge it is part of. The nodes (endpoints) are treated as free objects in space, fitting in with the semi-rigid body concept arising from considering the impact to any edge it is part of. As a result, we define the interior of the cell to be the domain to the left when traversing the boundary of the cell in an anti-clockwise direction. The force is then

$$\mathbf{F}_A = -F(\mathbf{r}_A) \cdot \mathbf{u} \mathbf{u},$$

where the dot product will evaluate to negative values when $J$ is inside the cell.

Applying a force to endpoints. As described above in the ‘Finding intersections’ section, the point of action is defined as the first point an expanding circle touches a given edge. Around an endpoint, this produces a curved region of interaction with a constant radius. Supplementary Fig. 4 shows the interaction region around an endpoint that is part of a cell boundary. Likewise, Supplementary Fig. 3a shows the curved region around the endpoints of an isolated edge. If a node falls into this part of the interaction region, the interaction force is applied directly to the endpoint. When this happens, we can calculate the force on $A$ in the same way as in equation (22):

$$\mathbf{F}_j = -F(\mathbf{r}_E) \cdot \mathbf{u} \mathbf{u},$$

as, by definition, in this case $A$ will lie on the endpoint.

The forces applied to a node are equal and opposite to the forces applied to an edge at the point of action; thus, the force applied to a node due to the edge is

$$\mathbf{F}_{ij} = -\mathbf{F}_{ji},$$

where $K$ is an edge that $i$ is not part of, and $A$ is the point of action on $K$ due to this node–edge interaction. Depending on the situation $\mathbf{F}_i$ may be defined by equation (22) or equation (24). This also applies to the opposite node in a node–node interaction calculated in equation (25).

In the models presented here, we resolve forces applied to a node without considering the impact to any edge it is part of. The nodes (endpoints) are treated as free objects in space, fitting in with the semi-rigid body concept arising from considering the impact to any edge it is part of. As a result, we define the interior of the cell to be the domain to the left when traversing the boundary of the cell in an anti-clockwise direction. The force is then

$$\mathbf{F}_A = -F(\mathbf{r}_A) \cdot \mathbf{u} \mathbf{u},$$

where the dot product will evaluate to negative values when $J$ is inside the cell.

Applying a force to endpoints. As described above in the ‘Finding intersections’ section, the point of action is defined as the first point an expanding circle touches a given edge. Around an endpoint, this produces a curved region of interaction with a constant radius. Supplementary Fig. 4 shows the interaction region around an endpoint that is part of a cell boundary. Likewise, Supplementary Fig. 3a shows the curved region around the endpoints of an isolated edge. If a node falls into this part of the interaction region, the interaction force is applied directly to the endpoint. When this happens, we can calculate the force on $A$ in the same way as in equation (22):

$$\mathbf{F}_j = -F(\mathbf{r}_E) \cdot \mathbf{u} \mathbf{u},$$

as, by definition, in this case $A$ will lie on the endpoint.

The forces applied to a node are equal and opposite to the forces applied to an edge at the point of action; thus, the force applied to a node due to the edge is

$$\mathbf{F}_{ij} = -\mathbf{F}_{ji},$$

where $K$ is an edge that $i$ is not part of, and $A$ is the point of action on $K$ due to this node–edge interaction. Depending on the situation $\mathbf{F}_i$ may be defined by equation (22) or equation (24). This also applies to the opposite node in a node–node interaction calculated in equation (25).

In the models presented here, we resolve forces applied to a node without considering the impact to any edge it is part of. The nodes (endpoints) are treated as free objects in space, fitting in with the semi-rigid body concept arising from considering the impact to any edge it is part of. As a result, we define the interior of the cell to be the domain to the left when traversing the boundary of the cell in an anti-clockwise direction. The force is then

$$\mathbf{F}_A = -F(\mathbf{r}_A) \cdot \mathbf{u} \mathbf{u},$$

where the dot product will evaluate to negative values when $J$ is inside the cell.
if there are fewer than six neighbors. Here $e_i$ is the distance to neighbor cell $i$, $n_i$ is the number of cells, $e_r$ is the distance to edge $r$, $m$ is the number of edges, $r$ is the resting cell radius, and $d_{\text{edge}} = \frac{1}{m} \sum_{i=1}^{m} e_i$. If there are no edge neighbors, then the center-based method is used.

Solving the equations of motion. Combining all of this together to calculate the motion of a node, we sum the directly applied forces, the rotation forces due to the edges, and the internal cell forces to give

$$F_j = F_{\text{Direct}} + F_{\text{Rotation}} + F_{\text{Cell}}$$

then apply the Forward Euler method to find its new position

$$r_j(t + \Delta t) = r_j(t) + \eta \frac{\Delta t}{m} F_j.$$  

Practical model implementation details. Beyond the calculation of forces that constitute the intended behavior of the model, there are practical decisions that need to be made in the modeling process and the implementation of the numerical library. These could be substantial additions to stop unphysical situations or small details that nevertheless could affect the tissue-level behavior observed.

Isolated edges require special care to make sure the simulation remains physical (that is, by stopping edges from crossing). Edges that are part of a polygon cells are no different. If an edge around a cell boundary is small, and the angle it makes with connected edges is also small, we can end up with edge inversion, where the boundary forms a non-simple polygon as shown in Supplementary Fig. 5. Small edges can also cause issues in applying the division process even with simple polygons.

One way to prevent edge inversion is to use a force based on the length of the edge as done in equation (28). This can be a simple linear spring that forces edges to a preferred natural length. When dealing with polygonal cells that can grow in size and divide, it is helpful to consider how edges contribute to the whole boundary. Here we implement a regularizing linear spring force which is applied to all edges around the boundary to keep the edge lengths relatively consistent. For a node $j$ that is part of edge $k$, both subsequently part of cell $M$, which is made up of $N_k$ edges and has perimeter of length $p_M$, we can define a regularizing force as

$$F_{\text{Reg}}^j = \sum_{k=1}^{N_k} -\alpha_{\text{reg}} (l_k - p_M/N_k) u_{jk},$$

In summary, each edge around a cell experiences a force so that they all tend toward the same length, being an equal fraction of the current measured perimeter. We note here that this force has no physical basis, and it is used exclusively to equalize the lengths of the edges.

It is not always possible to prevent edges from becoming too small, even with the regularizing force, and in fact sometimes we may want to allow small edges. To prevent inversion in these cases we target the formation of small angles by using the node–edge interaction mechanism to repel edges from the same cell. To invert, an endpoint needs to cross an edge from the same polygon, so we introduce a small regularizing force based on the length of the edge as done in equation (28). This can be a simple linear spring that forces edges to a preferred natural length. When dealing with polygonal cells that can grow in size and divide, it is helpful to consider how edges contribute to the whole boundary. Here we implement a regularizing linear spring force which is applied to all edges around the boundary to keep the edge lengths relatively consistent. For a node $j$ that is part of edge $k$, both subsequently part of cell $M$, which is made up of $N_k$ edges and has perimeter of length $p_M$, we can define a regularizing force as

$$F_{\text{Reg}}^j = \sum_{k=1}^{N_k} -\alpha_{\text{reg}} (l_k - p_M/N_k) u_{jk},$$

References

1. Huang, M., Garbery, M., Berceli, S. A. & Tran-Son-Tay, R. Rule-based simulation of multi-cellular biological systems—a review of modeling techniques. *Cell. Mol. Bioeng.* 2, 285–294 (2009).

2. Kari, J., Herrmann, M., Peyriéras, N. & Doursat, R. A cell-based computational model of early embryogenesis coupling mechanical behaviour and gene regulation. *Nat. Commun.* 8, 13929 (2017).

3. Van Leeuwen, I. M. et al. An integrative computational model for intestinal tissue renewal. *Cell. Biol.* 42, 617–636 (2009).

4. Odell, G. M., Oster, G., Alberch, P. & Burdsale, B. The mechanical basis of morphogenesis I: epithelial folding and invagination. *Dev. Biol.* 85, 446–462 (1981).

5. Drasdo, D. & Hühme, S. Individual-based approaches to birth and death in vasculogenic tumors. *Math. Comput. Model.* 37, 1163–1175 (2003).

6. Bull, J. A., Mech, F., Quaiser, T., Waters, S. L. & Byrne, H. M. Mathematical modelling reveals cellular dynamics within tumour spheroids. *PLoS Comput. Biol.* 16, e1007961 (2020).

7. Meinke, F. A., Potten, C. S. & Loeffler, M. Cell migration and organization in the intestinal crypt using a lattice-free model. *Cell. Biol.* 34, 253–266 (2001).

8. Nagai, T. & Honda, H. A dynamic cell model for the formation of epithelial tissues. *Phil. Mag. B* 81, 719–742 (2001).

9. Okuda, S., Inoue, Y. & Adachi, T. Three-dimensional vertex model for simulating multicellular morphogenesis. *Biophys. Physicobiol.* 12, 13–20 (2015).

10. Newman, T. J. Modeling multicellular systems using sub-cellular elements. *Math. Biosci.* 2, 613–624 (2005).

11. Merks, R. M., Gurvage, M., Inzé, D. & Beemster, G. T. Virtuelleaf: an open-source framework for cell-based modeling of plant tissue growth and development. *Plant Physiol.* 155, 668–671 (2011).

12. Osborne, J. M., Fletcher, A. G., Pitt-Francis, J. M., Maini, P. K. & Gavaghan, D. J. Comparing individual-based approaches to modelling the self-organization of multicellular tissues. *PLoS Comput. Biol.* 13, e1005387 (2017).

13. Okuda, S., Miura, T., Inoue, Y., Adachi, T. & Eiraku, M. Combining turing and 3D vertex models reproduces autonomous multicellular morphogenesis with undulation, tubulation, and branching. *Sci. Rep.* 8, 2386 (2018).

14. Durand, R. E. Review revascularization strategies as a model for cell kinetic studies. *Cell. Biol.* 3, 141–159 (1990).

15. Karolak, A., Markow, D. A., McCawley, L. J. & Reinjatk, K. A. towards personalized computational oncology: from spatial models of tumour spheroids, to organoids, to tissues. *J. R. Soc. Interface* 15, 20170703 (2018).

16. Harding, M. J., McGraw, H. F. & Nechiporuk, A. The roles and regulation of multicellular rosette structures during morphogenesis. *Development* 141, 2549–2558 (2014).

17. Fletcher, A. G., Osterfield, M., Baker, R. E. & Sharpmans, S. Y. Vertex models of epithelial morphogenesis. *Biophys. J.* 106, 2291–2304 (2014).

18. Merouzou, A., Malaspintas, O., Trushko, A., Roux, A. & Chapard, B. Influence of cell mechanics and proliferation on the buckling of simulated tissues using a pressure model. *Nat. Comput.* 17, 519–519 (2018).

19. Jiménez, J. J., Segura, R. J. & Feito, F. R. Efficient collision detection between 2D polygons. *J. WSCG* 12, 191–198 (2004).
20. Alnet, A. A., Maini, P. K., Moulton, D. E. & Byrne, H. M. Modeling perspectives on the intestinal crypt, a canonical system for growth, mechanics, and remodeling. *Curr. Opin. Biomed. Eng.* 15, 32–39 (2020).

21. Volkson, D., Cookson, S., Hasty, J. & Tsai, M. Biomechanical ordering of dense cell populations. *Proc. Natl Acad. Sci. USA* 105, 15346–15351 (2008).

22. Doi, M. & Edwards, S. F. *The Theory of Polymer Dynamics* Vol. 73 (Oxford Univ. Press, 1988).

23. Rudge, T. J., Steiner, P. J., Phillips, A. & Haseloff, J. Computational modeling of synthetic microbial biofilms. *ACS Synth. Biol.* 1, 345–352 (2012).

24. Norton, K.-A. et al. A 3D mechanistic model of breast ductal carcinoma in situ (DCIS) morphogenesis and progression. *J. Theor. Biol.* 263, 393–406 (2010).

25. Buske, P., Przybilla, J., Leefler, M. & Galle, J. The intestinal stem cell niche: a computational tissue approach. *Biochem Soc Trans.* 42, 671–677 (2014).

26. Venugopalan, G. et al. Multicellular architecture of malignant breast epithelia influences mechanics. *PLoS ONE* 9, e101955 (2014).

27. Metzcar, J., Wang, Y., Heiland, R. & Macklin, P. A review of cell-based computational modeling in cancer biology. *JCO Clin. Cancer Inform.* 2, 1–13 (2019).

28. Sengupta, N. & MacDonald, T. The role of matrix metalloproteinases in stromal/epithelial interactions in the gut. *Physiology* 22, 401–409 (2007).

29. Schoenwolf, G. C. & Smith, J. L. Mechanisms of neurulation: traditional viewpoint and recent advances. *Development* 109, 243–270 (1990).

30. Humphries, A. & Wright, N. A. Colonic crypt organization and tumorigenesis. *Nat. Rev. Cancer* 8, 415–424 (2008).

31. Honda, H. Description of cellular patterns by dirichlet domains: the two-dimensional case. *J. Theor. Biol.* 72, 523–543 (1978).

32. Bentley, J. L., Stanat, D. F. & Williams, E. H. Jr. The complexity of finding fixed-radius near neighbors. *Inf. Process. Lett.* 6, 209–212 (1977).

33. Mirams, G. R. et al. Chaste: an open source C++ library for computational physiology and biology. *PLoS Comput. Biol.* 9, e1002970 (2013).

34. Ghaffarizadeh, A., Heiland, R., Friedman, S. H., Mumenthaler, S. M. & Macklin, P. PhysiCell: an open source physics-based cell simulator for 3D multicellular systems. *PLoS Comput. Biol.* 14, e1005991 (2018).

35. Alarcon, T., Byrne, H. & Maini, P. Towards whole-organ modelling of tumour growth. *Prog. Biophys. Mol. Biol.* 85, 451–472 (2004).

36. Borle, A. B. Kinetic analyses of calcium movements in hela cell cultures: I. Calcium influx. *J. Gen. Physiol.* 53, 43–56 (1969).

37. Posakony, J. W., England, J. M. & Attardi, G. Mitochondrial growth and division during the cell cycle in hela cells. *J. Cell Biol.* 74, 468–491 (1977).

38. Guillot, C. & Lecuit, T. Mechanics of epithelial tissue homeostasis and morphogenesis. *Science* 340, 1185–1189 (2013).

39. Dunn, S.-J. et al. A two-dimensional model of the colonic crypt accounting for the role of the basement membrane and pericryptal fibroblast sheath. *PLoS Comput. Biol.* 8, e1002515 (2012).

40. Langlands, A. J. et al. Paneth cell-rich regions separated by a cluster of Lgr5+ cells initiate crypt fission in the intestinal stem cell niche. *PLoS Biol.* 14, e1002491 (2016).

41. Paulsson, M. Basement membrane proteins: structure, assembly, and cellular interactions. *Crit. Rev. Biochem. Mol. Biol.* 27, 93–127 (1992).

42. Macklin, P., Edgerton, M. E., Thompson, A. M. & Cristini, V. Patient-calibrated agent-based modeling of ductal carcinoma in situ (DCIS): from microscopic measurements to macroscopic predictions of clinical progression. *J. Theor. Biol.* 301, 122–140 (2012).

43. Dunn, S.-J., Nätke, I. S. & Osborne, J. M. Computational models reveal a passive mechanism for cell migration in the crypt. *PLoS ONE* 8, e80516 (2013).

44. Meri, A. & Bra, L. Engineering Mechanics: Statics 4th edn (Wiley, 2003).

45. luckyphill. luckyphill/EdgeBased: Rigid body framework paper Rev1 Version 1.0.1. https://doi.org/10.5281/zenodo.4817386 (2021).

Acknowledgements

P.J.B. acknowledges J. Krokiewski for his extremely thorough text on Mechanics of a Rigid Body, which was provided free of charge to students of mechanical engineering at The University of Melbourne, without which this work would not have been possible. This work was supported with supercomputing resources provided by the Phoenix HPC service at the University of Adelaide. B.J.B. and P.J.B. acknowledge funding from the ARC (grant number DP160102644).

Author contributions

P.J.B. conceived and developed the framework, developed the software, ran the simulations, and performed the analysis. All authors designed the exemplar models. P.J.B. and J.M.O. wrote the manuscripts; J.E.G., B.J.B. and J.M.O. supervised the project. All authors read and approved the final manuscript.

Competing interests

The authors declare no competing interests.

Additional information

Supplementary information The online version contains supplementary material available at https://doi.org/10.1038/s43588-021-00154-4.

Correspondence and requests for materials should be addressed to Philip J. Brown, J. Edward F. Green, Benjamin J. Binder or James M. Osborne.

Peer review information *Nature Computational Science* thanks Paul Macklin and the other, anonymous, reviewer(s) for their contribution to the peer review of this work. Handling editor: Ananya Rastogi, in collaboration with the *Nature Computational Science* team.

Reprints and permissions information is available at www.nature.com/reprints.

Publisher’s note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

© The Author(s), under exclusive licence to Springer Nature America, Inc. 2021, corrected publication 2022.