A Human-in-the-Loop Environment for Developmental Biology

Melanie Däschinger[1], Andreas Knote[2], Rebecca Green[3] and Sebastian von Mammen[2]

[1]Universität Augsburg, Germany
melanie.roswitha.daeschinger@student.uni-augsburg.de
[2]Julius-Maximilians-Universität Würzburg, Germany
{andreas.knote,sebastian.von.mammen}@uni-wuerzburg.de
[3]University of Calgary, Canada
rgree@ucalgary.ca

Abstract

BOODLE (BiOlOgical DeveLopment Environment) is a long-term project to complement morphometric empirical studies in the field of developmental biology by means of interactive modelling and simulation techniques. BOODLE aims at providing viable behaviour models of cells that fit recorded time series of morphogenetic stages. This information is critical for driving empirical studies and for simulating the emergence of pathological abnormalities. In this paper, we present a BOODLE prototype that covers the whole functionality of the envisioned system. The application usage cycle starts with importing scan data of embryos of model organisms. Next, it allows the user/experimenter to work with the imported model, for instance to add meta data, to refine annotations, and even to automatically or manually populate the captured volumes with virtual cells. Finally, the user/experimenter is given the opportunity to run simulation experiments. To outline how the pipeline works in particular, we have setup a mockup toy experiment that optimises cell parameters such that a population of cells develops in accordance with several preset transient states. For all stages of the modelling and simulation process, BOODLE provides accessible interfaces and visualisations. These include visual programming and configuration of individual cells' behaviours and physical properties. In this paper, we show how all these aspects together realise a prototype for a Biologist-in-the-Loop simulation for the creation, automated optimisation and analysis of cellular behaviour models.

Introduction

Ever-growing capabilities of modern microprocessors, vast improvements in multi-core system designs and their availability as well as raw computing powers offered by cloud services and according algorithmic solutions are becoming ever more accessible. Computational simulation experiments bear several important advantages over lab work. They can shed light on critical modelling issues, for instance, by providing the means to run comparisons between alternative hypotheses, by optimising sets of model parameters, or even by generating whole new hypotheses for empirically captured data. They can be run over long periods of time, arbitrarily re-run, easily adjusted and their realisation is comparably cheap. The ability to interact directly with a subject of scientific interest often can provide insight and understanding that requires far more abstraction work when it has to be obtained through retrospective, static analysis. Such interactive simulations need to be fed by raw power, and their flexibility often requires compromise in accuracy or model complexity. We feel that the gains obtained from real time interactive in-silico laboratories can be substantial. However, next to the great demand for computing power, a multitude of complex computer science challenges obstructs the path to harnessing these benefits. Computational models need to be found that capture different kinds of interactions and at different levels of abstraction. An according infrastructure needs to be provided that can integrate empirically compiled data, both structural and behavioural, according multi-physics models, algorithmic entanglement of multiple scales and computational representations of different specificities. Once these aspects are in place, efficient heuristic optimisation approaches, such as evolutionary algorithms, can serve to generate and hone fitting behavioural models. Apt user interfaces and visualisations need to be in place to render the envisioned computing framework accessible. They need to facilitate model building and verification by domain experts from the life sciences.

In this paper, in analogy to Human-in-the-Loop simulations (Narayanan and Kidambi, 2011), we present our approach to an according Biologist-in-the-Loop system for interactive modelling and simulation of biological developmental processes (BOODLE). It provides a vast range of user-centred functionalities tailored specifically to support work by developmental biologists. Its primary goal is to support them in testing model hypotheses but also, in the longer run, to have the BOODLE system generate solutions to hard problems such as suggesting plausible cell behaviours that retrace empirically observable phenomena. To these ends, BOODLE not only makes it possible to import and visualise CT data at interactive speeds but also to proactively work with this data, for instance to label certain areas or subvolumes, or to define optimisation constraints that are then considered by the aforementioned optimisation engines. For the prototype presented in this paper we placed great empha-

sis on the accessibility of the functionality of various stages of the BOODLE application usage cycle. Hence, we have realised according visualisations and interaction tasks.

The remainder of this work is structured as follows. In the next section, we present seminal related works regarding cell-based simulations, technologies that BOODLE has been built upon, and the inherent challenge of guiding self-organising processes, which is essential for generating and honing computational models that are aligned with empirically retrieved data. Afterwards, we present the design, functionalities and workflows of BOODLE, whereas a concrete example of its intended use is demonstrated. We conclude with a summary and an outline of potential future work.

Related Work

There have been numerous computational models of complex cellular behaviours. However, the majority of the existing models are often either overly abstracted (von Mammen et al., 2012) or specialised (Santos et al., 2004) for the simulation of tissue formation. In fact, with respect to developmental biological models and simulations, especially the importance of physical interactions has recently been stressed. For example, the formation of supply networks or the growth and differentiation of plant stems cannot sufficiently be retraced without considering physical stresses (Drasdo et al., 2007; Hamant et al., 2008; Uyttewaal et al., 2010). Therefore, computational models of developmental processes ideally account for biological as well as for physical interactions. An example is provided by (Disset et al., 2014), where cells are modelled as elastic spherical bodies. The cells can establish fine-tuned adhesive forces among each other, which can break based on external stresses, and which serve as a basis for forming tissue layers. In general, it is important that cell-centered biological developmental models retrace the fundamental interactions listed by (Salazar-Ciudad et al., 2003). They include division, induction, adhesion, apoptosis, migration, contraction and matrix modification (swelling, decomposition, or loss).

CellSys (Swat et al., 2012) is an exemplary framework for modelling and simulating developmental processes, and also provides components for visualisation and analysis. Again, each cell is represented as a spherical, elastic body that can divide, grow, and migrate. Deformation, compression and adhesion are implemented following the Johnson-Kendall-Roberts (JKR) model, which defines the contact mechanics between elastic spheres. It has been proven that the JKR model applies to the domain of living cells for as long as the cytoskeletons are not disrupted (Chu et al., 2005). Explicit Euler integration drives the equations for diffusion across discrete grids and consumption of nutrients and growth factors. Cell parameters that CellSys supports include, for instance, their elasticity, diameter, the diffusion rate of morphogens, surface adhesion, initial orientation of cells. As a visual outcome of a simulation, the modeller can specify the colour spectra to highlight areas of pressure, contact, growth and deformation within the simulated cell populations. In addition, visualised components can be filtered based on attributes such as the activity or orientation of cells. CellSys employs several third-party libraries and interfaces such as OpenGL, PovRay, OpenMP (in order to parallelise algorithms), SuperLU (in order to parallelise calculations in the context of shared-memory systems) and GLUI (for the development of graphical user interfaces).

Based on these deliberations and preceding works, we have decided to design BOODLE to integrate established and continuously extended and improved third-party libraries. Beyond the fundamental features introduced above, our work strives for real-time interactivity. To achieve a more fine-grained, real time interactive physical cell model, we use the particle-based physics engine FLEX (Macklin et al., 2014). FLEX considers all physical objects of a simulation to be composed of numerous particles. Local constraints among the particles determine the global physical properties of the resultant objects. This approach allows one to consider soft bodies, deformable bodies, rigid bodies, and fluids within a single simulation context. FLEX makes heavy use of GPUs and there are several techniques in place to improve the robustness of physical calculations. Furthermore, it is subject to ongoing development efforts.

In order to efficiently import and work with volumetric data sets, we rely on the Point Cloud Library (PCL) (Aldoma et al., 2012). The technologies we utilise all support real-time calculations or may be run offline to efficiently fetch and make use of their results while the simulation is running. In terms of BOODLE's front end, we rely on different visualisation techniques pursuing novelty, informativeness, efficiency and aesthetics (Steele and Iliinsky, 2010) and to present the imported voxel data in 3D space (Jeong et al., 2010). Next to third-party libraries that provide functionalities for user interface designs and modalities, visualisation techniques, data preprocessing steps and rendering pipelines, BOODLE also needs to address the grand challenge of approximating empirically identified structures and constraints based on individual cellular behaviours.

In artificial life research, this challenge coined the term guided self-organisation, or GSO. Generally speaking, GSO aims at models that result in a well-directed increase of organisational structure or functionality without providing explicit instructions. Until today, self-organising construction is a prospering field of research, see, e.g. (Werfel et al., 2014; Napp and Nagpal, 2014; Soleymani et al., 2015). It has been broadly acknowledged that guiding self-organisation is an important concept to mastering complex systems. An overview of guided self-organisation is provided by (Prokopenko, 2014). GSO draws from formal methods to describe and effectively guide self-organising processes (Prokopenko et al., 2014). To this end, measures of organ-

isation and complexity play a crucial role, as do means to capture the sensitivity of self-organising systems to various inputs (Prokopenko et al., 2015). Along these lines, empowerment, which expresses an agent's means to handle different situations is another important measure to inform GSO (Salge et al., 2014). In general, tracking, understanding and predicting the flow of information in self-organising systems are the foundations to successfully guiding them. Ay and Zahedi (2014), for instance, trace the flow between agents' sensors and their motor activities and draw conclusions about the system behaviour probabilisitically. Alternatively, one can influence a self-organising system by setting constraints or by means of functionals (Gros, 2014) which define general goals rather than clear-cut target states relying on probability functions. Any attempts to let bottom-up calculations and top-down specifications converge can be pursued to implement an effective GSO. Interactive evolutionary computation (IEC) realises this idea very directly. Simulations are run bottom-up, the results are presented to and evaluated by an operator in such that globally desirable features are preferred. As an example, an IEC was presented to breed swarm chemistries, in which swarms of particles are configured to exhibit different movement patterns and spatial formations (Sayama, 2014). Taking interaction out of the loop, evolutionary computing itself can also guide self-organising processes as, for instance, presented by (Lobo and Levin, 2015). Here, inner-cellular gene expression pathways in tandem with parameters concerning inter-cellular communication were optimised to retrace processes of tissue regeneration.

Design and Functionality of BOODLE

In this section, we first lay out the cell model that we have currently integrated into BOODLE. Second, we elaborate on its functionality with respect to data imports, editing and organisation.

The Cell Model

Like any simulator, accuracy and efficiency are both fundamental factors in BOODLE's success. Hence, we are continuously improving our model of the virtual cells. Currently, our virtual cell is represented as a spherical mesh surface covered with 64 virtual physical particles as seen in Figure 1. Different scenarios may require for more or less fine grained representations, which can be achieved by changing the number of particles used. As can be seen in Figure 1, the constraints among the physics particles determine, among other aspects, the sizes of the cells. A mesh is wrapped around the particles for visualisation purposes. Different from approaches where the mesh itself would encode the physical shape of an object, such a particle-based approach provides for a far more flexible solution as the particle constraints can be changed during the simulations.
Cohesion, division and diffusion of the virtual cells are determined by according parameters, which can be defined by the modeller utilising visual programming scripts called Blueprints (Figure 2). Blueprints are a part of the Unreal Engine, a development environment for interactive realtime systems that we use for content integration and interactive visualisation.

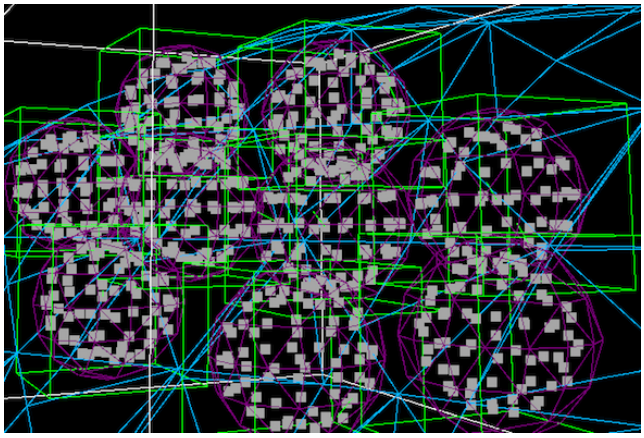


Figure 1: Nine virtual cells, each of which is comprised of 64 physics particles. The constraints between that hold the particles in a spherical shape can be adjusted during the simulation, for instance to simulate cell growth.

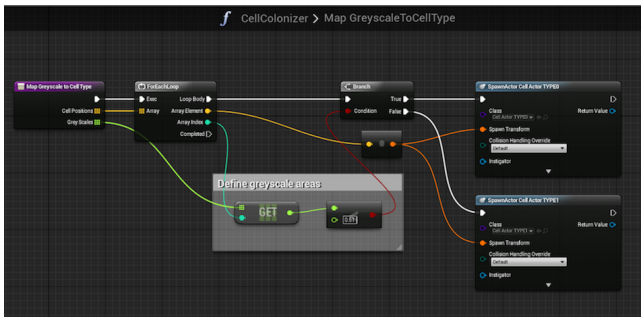


Figure 2: A visual programming Blueprint that determines the population of embryonic volumetric data based on grey values.

Figure 3 shows an experiment where aspects of tissue maintenance under stress is simulated. A spherical rigid body (pink) is populated on its surface by virtual cells which adhere to each other and divide if the tensile stress they experience exceeds a certain limit. The cells were not configured to actively fill void spaces. This is a basic example of the capabilities of combined soft and rigid body physics possible with our approach.

BOODLE Workflows

A big part of the work conducted by developmental biologists is characterised by taking measures of morphologic structures in combination with the underlying gene expres-

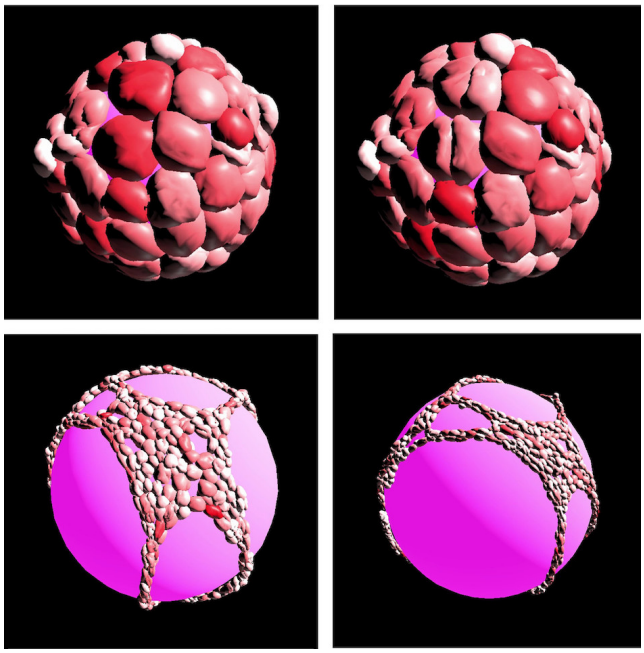


Figure 3: Simulated cells, based on a variant of our cell model, are subjected to tensile stress in an early experiment. Images ordered in sequence from top-left to bottom right. Tissue cells, colored from white to red depending on the experienced tensile stress, placed on a spherical body (pink), divide when the tensile stress exceeds a set threshold. The sphere increases its volume continuously, whereas the cells maintain a low stress level by dividing accordingly.

sions (Zelditch et al., 2012) and inferring according, general laws, e.g. (Xu et al., 2015). The basis for these inquiries is provided by series of volumetric data sets that capture model organisms such as mice and chicks at different developmental stages (Schambach et al., 2010). In order to accommodate this empiric methodology, BOODLE allows to import CT scan data and meta data into the simulation contexts, to mark surfaces and volumes, to populate specific areas with virtual cells and to observe, measure and log simulated model alterations.

In Figures 4 and 5, the main menu is visible at the top of the screen, offering access to the simulation's main features. From left to right, it allows to load CT scan data (*Import*), place and modify landmarks (*Landmarks*), populate the simulation with virtual cells (*Cells*), start end stop the simulation (*Simulate*) and save the simulation state (*Save*).

Importing Volumetric Data We can import CT scan data as point clouds into BOODLE. The raw data is used to define the simulation model as volumetric data points (voxels), and an additional mesh-based visual representation is generated from it. Here, PCL (see the Section on Related Work) provides numerous efficient algorithms used in the preprocessing, e.g. for segmenting volumetric data, filtering, mesh generation or model adaption. The voxel data then allows to automatise the process of populating large regions of the model space with specifically configured, virtual cells. Furthermore, the user can import full series of CT data sets and order them with respect to the lifetime of the scanned specimen. Imported, pre-processed data is stored to ensure that the time-consuming pre-processing step of the volumetric data (including filtering, removal of outliers, and downsampling) only has to be performed once.

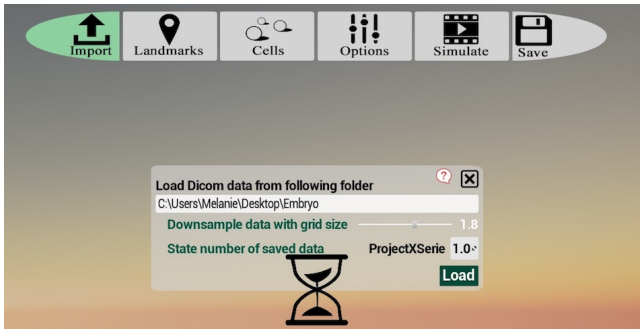


Figure 4: User interface dialogue for importing a volumetric data set to be considered in the context of a specific simulation state.

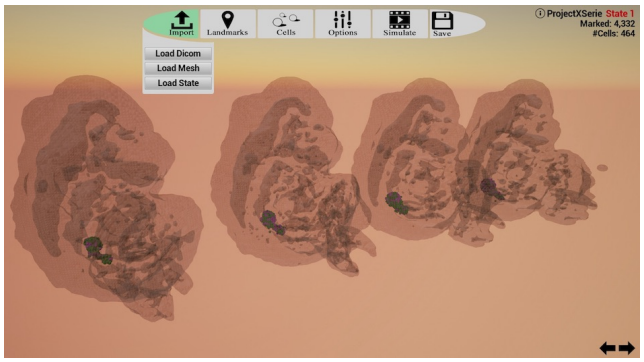
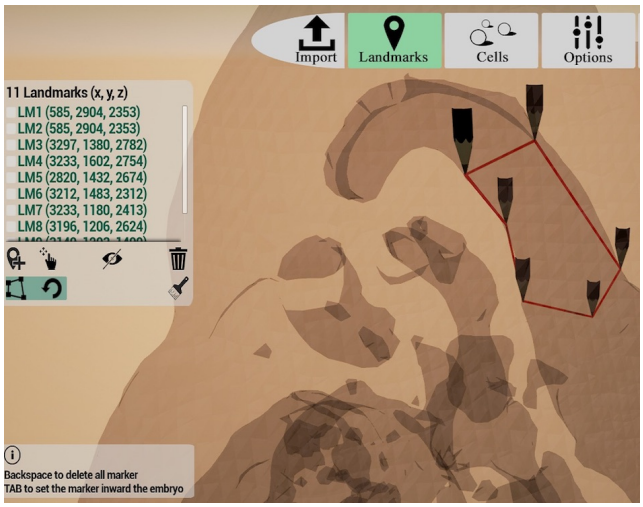
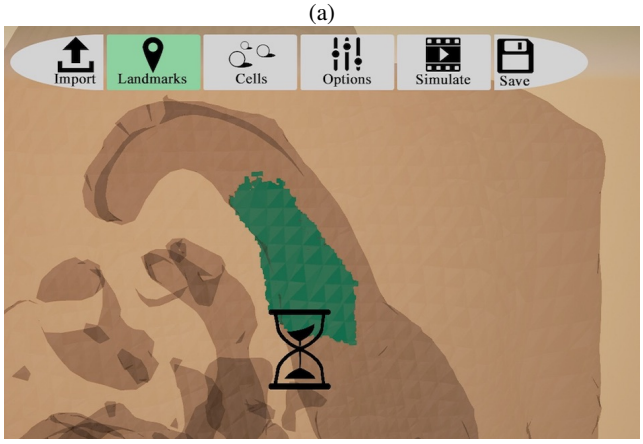


Figure 5: Visualisation of four simulation states that comprise the volumetric data from individual CT scans, metadata such as landmarks as well as data about the involved virtual cells.

The import process is controlled from the main menu (see Figure 4). Beyond specifying the path to the data set, only two parameters are exposed to the user: Determining (a) the *rate of down-sampling* of the volumetric data set (value range between 1.2 and 2.2), and (b) the *association with and the order within* a specific import series (project name and index). We refer to a single imported data set as a specific *simulation state*. A time series of simulation states is exemplarily shown in Figure 5. Landmark data, as well as the data to define and initialise initial populations of virtual cells, is stored at the granularity of such simulation states.



(a)



(b)

Figure 6: The user can mark areas by (a) drawing polygons and (b) select the voxels underlying these areas.

Editing Landmarks In order to effectively trace the cellular developments throughout an experiment, biologists segment and mark certain areas of the scanned data by means of morphometric software toolkits. They pin landmarks at specific spatial locations. As an example, consider the work by Hu et al. who have successfully traced and formalised the crano-facial development in chick embryos (Hu et al., 2003; Hu and Marcucio, 2009). Therefore, and for the purpose of evolutionary optimisation, our environment allows the user to import according landmark data sets, to set them up in the context of the simulation, to remove, edit and store them, and to activate/deactivate their visualisation. It further provides the means to translate one or two-dimensional landmarks into spatial, three-dimensional selections of the underlying voxel data. More specifically, individual landmark points can be augmented by local spherical volumes to select voxel data. To this end, the user can adjust the selection sphere's radius. Alternatively, the user can determine that exactly the k-nearest voxels to a landmark should be selected, again leaving only one parameter to be adjusted by the user. The most generic way that BOODLE offers is voxel selection based on polygons. This is realised by sequentially drawing lines to place new pins until the first pin is re-selected. Then, the polygon is completed and all the voxels captured are visualised as seen in Figure 6.

Populating Volumes with Virtual Cells Based on the selected volumes or voxels, the user may introduce virtual cells into a simulation state. Beyond manually placing cells individually by clicking on the targeted voxels, he can populate large volumes automatically. If desired, he may even introduce a distinction regarding the cell type to occupy voxels of different grey values. The user can adjust the respective thresholds and introduce new types of cell configurations by means of the Blueprint script shown in Figure 2. The automated population of volumes with virtual cells takes place as follows. First, a random voxel within the target set is chosen. Second, the chosen voxel is assigned all those neighbouring voxels that are closer or equal to a given cell radius parameter. The average grey value of all the selected voxels is then considered for initialising a virtual cell of a specific, pre-defined type. Its initialisation location equals the geometric centre of mass of the selected voxels. A mix of manual and automated population ensures that the modellers can design, control and edit large numbers of virtual cells and align their initialisation with empirically retrieved volumetric data. A domain expert can extend the cells' behaviours based on Blueprint programmes as illustrated in Section III and also deploy different cell types in a single simulation scenario.

Guided Self-Organisation The promise of interactive biological simulation frameworks such as BOODLE is (a) the inexpensive, automated and incessant search for accurate and complete models that backup empirically retrieved data, (b) the discovery of misalignments in theories and of those aspects that need to be investigated most urgently, and, in the long run, (c) the comprehensive prediction of biological development and evolution at micro and macro scales. Albeit important, the integration of volumetric data, making it accessible and interactively editable is far from fulfilling these ambitious goals. In order to reach them, small, careful steps need to be taken first. For now, we resort to a heuristic optimisation technique called genetic algorithms (Goldberg and Holland, 1988). Inspired by evolutionary biology, the overarching class of evolutionary optimisation approaches makes use of the concepts of fitness-based selection, mutation and recombination to breed a pool of well-adapted solutions. To account for time series data as found in sequential CT scans, we integrated time-dependency into the genetic algorithm similar to (von Mammen and Däschinger, 2015). The configuration of a simulation state's cell population represents a single solution and so-called target states

at predefined simulation time steps determine the fitness of a solution by computing the difference between the achieved configuration and the target configuration.

Approximation of Tissue Growth

In order to demonstrate the potential workflow of the BOODLE framework, we created a small mockup example based on a volumetric data set that we retrieved from a chick embryo. The overview of this experiment is presented in (Däschinger et al., 2017). Currently, the embryonic data set only serves to illustrate the work pipeline, as we have not imported and worked with comprehensive development series, yet. The mockup experiments show how the user/experimenter may utilise the BOODLE framework to optimise cell models which, starting from an initial population, allow to retrace the development of specific tissue regions defined by annotated CT data over several time steps. For each one out of four target states, we manually introduced changes in the shape and the spatial dimensions of four independent target surfaces. Afterwards, we populated the initial surfaces with two types of virtual cells (purple and green) as seen in Figure 7.

Cell Representation

The cells' configuration space is based on the model presented above, whereas both their biological and their physical behaviours are fully determined by sets of parameter values as opposed to complex interaction rules. We divide the cells' genotype in alleles of three main categories: (1) Cohesion, (2) division, (3) chemical communication. Cohesion-specific alleles determine: The minimal distance between two cells to stick together (cohesion distance), the stiffness/deformability of a cell (cell stiffness), and the maximal number of sticking neighbours of a cell (max. clusters). Division-specific alleles specify: Whether a cell divides based on physical stresses from its neighbours (stress-based division), the respective stress level that triggers division, and whether such stress-based division applies to both involved cells (bi-direction division). Alleles revolving around chemical communication determine: Whether cells divide based on chemical signals (chemically-induced division), the minimal chemical value to trigger division (division signal strength), whether a cell reacts to morphogens (morphogen reactivity), the rate of morphogen signal emission of a cell per simulation tick (morphogen emission rate), and the way the cell reacts to morphogen gradients in its environment (morphogen reaction).

Table 1 lists all the considered variables that define the genotype of a cell, including parameter ranges. They either determine the activation/deactivation of the respective ability, or represent numeric intervals that we have empirically determined to support the respective interactions. Overall, the genotype is comprised of 12 parameters and 40 bits. For the given experiments, we omitted the cells' means of morphogenetic communication as the domain experts were interested to see shape and dimensions be retraced without it.

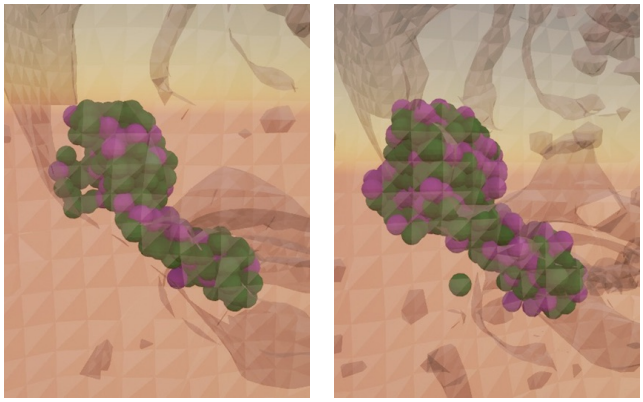
Parameter	*Range*
Cohesion distance	$[54; 85]$
Cohesion threshold	$[43; 58]$
Cell stiffness	$[0.01; 1.0]$
Max. clusters	$[1; 16]$
Stress-based division	$true/false$
Stress level	$[1.2; 1.5]$
Bi-directional division	$true/false$
Chemically-induced division	$true/false$
Division signal strength	$[5; 36]$
Morphogen emission rate	$[0; 127]$
Morphogen reactivity	$true/false$
Morphogen reaction	$attraction/repulsion$

Table 1: Parameter ranges of the alleles that determine the behaviour of a cell of a specific type.

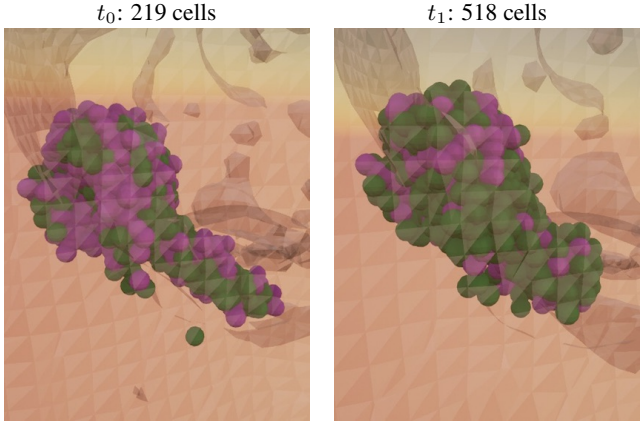
Experiment: Setup & Results

Our experiment defines several target states that are considered about $12h$ apart (actual time). The average time between the target states was compressed to 20 simulation seconds. For each target state, we defined a measure for the difference between the target volume's convex hull and the convex hull of the simulated volume based on a random sample of points on the two hulls. Lower differences imply better approximations of the target volumes, and, as a consequence, better fitnesses of the respective solutions. Let us point out, though, that this is only one potential dimension for calculating fitness values. Factors such as structural properties of emerging tissues, the ordering of cells, or a multitude of additional constraints could define additional fitness criteria. Not only is our first experiment limited in terms of fitness criteria, we also constrain ourselves in terms of the size of the targeted area and focus only on a small part of the embryo.

Figure 8 exemplarily shows four rather different morphologies at the last evaluation step t_3. The shortest distances (green lines) between the convex hull of the morphologies and the target volumes (red) are highlighted. The first phenotype (a) exhibited the lowest fitness of the four examples. Here, the total distance to the target morphologies increased over time as the simulated cells kept contracting and did not proliferate at all. The second phenotype (b) performed better; Here, the cells expanded in all directions. Despite the lack of proliferation, the target volumes were approximated more closely. Phenotype (c) reached a similar number of cells as the target state (1187 out of 1296). Yet, the generated cells migrated in the wrong direction which diminished its success. The best fitness was achieved by (d), but it required a rather large amount of simulated cells.

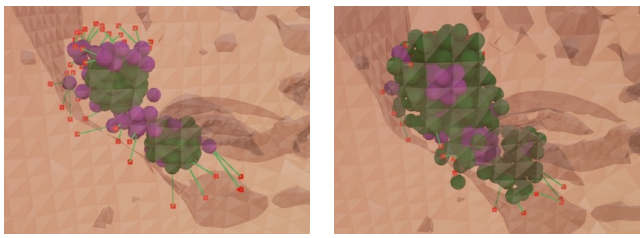


t_0: 219 cells t_1: 518 cells

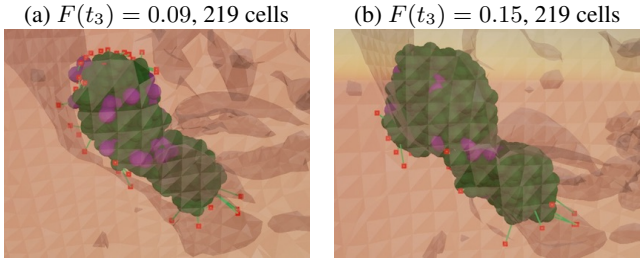


t_2: 889 cells t_3: 1296 cells

Figure 7: A series of four pre-defined target states at fixed points in time.



(a) $F(t_3) = 0.09$, 219 cells (b) $F(t_3) = 0.15$, 219 cells



(c) $F(t_3) = 0.17$, 1187 cells (d) $F(t_3) = 0.19$, 8392 cells

Figure 8: Different initial cell configurations resulted in different numbers of cells and different fitness values at t_3.

Conclusion

Despite its early state, BOODLE, a framework for interactive modelling and simulation tailored to the needs of developmental biologists, covers a comprehensive set of functionality. It allows the user/experimenter to import and annotate embryonic morphologies retrieved by means of CT scans. Landmarks and annotations introduced by external softwares are automatically adapted, additional information can be added. The CT data can, in principle, be utilised to stage biologically relevant experiments, for instance to generate or optimise model hypotheses that retrace empirically observable phenomena. The volumetric voxel information from CT data is maintained to inform the model building processes and mesh surfaces are generated for visualisation purposes. In addition, BOODLE allows the researcher to make use of landmark data for automated initialisation of in-silico cell populations. The transition from point locations and surface areas to the selection of volumetric voxels is facilitated by means of adjusting the dimensions of selection volumes, the reach of nearest-neighbour searches, or by manually drawing polygons to select voxel subsets. Storage and retrieval of series of simulation states, including different volumetric data sets as well as cell configurations, represent the foundation for tracing and re-engineering developmental processes over time. In combination, these functionalities and their respective interfaces empower the user to setup and run simulation and optimisation experiments. In addition to introducing BOODLE and detailing the different interaction tasks that define the usage cycle of the application, we presented a small mockup toy problem. It is meant to show how the components of the BOODLE framework can contribute towards supporting biologists in their efforts of model refinement. To this end, we deployed a genetic algorithm that identified configurations of cell clusters to let them successfully approximate four morphological target states.

There are four different research directions that need attention. First, the underlying simulation technologies have to be improved—in terms of efficiency and accuracy. Second, the user interfaces need to be systematically evaluated with respect to user experience and usability and honed accordingly. Third, large and diverse data sets of developmental time series need to be prepared for BOODLE and utilised. Only then, we will be able to discover shortcomings of the data processing and manipulation pipeline. Fourth, in order to address relevant research questions of developmental biologists, optimisers, such as the genetic algorithm deployed in the mockup experiment in this paper, need to be designed and adapted more rigorously, potentially also harnessing calculations that run on high-throughput hardware such as GPUs or distributed across computer clusters.

References

Aldoma, A., Marton, Z.-C., Tombari, F., Wohlkinger, W., Potthast, C., Zeisl, B., Rusu, R. B., Gedikli, S., and Vincze, M. (2012). Point cloud library. *IEEE Robotics & Automation Magazine*, 1070(9932/12).

Ay, N. and Zahedi, K. (2014). On the causal structure of the sensorimotor loop. In *Guided Self-Organization: Inception*, pages 261–294. Springer.

Chu, Y.-S., Dufour, S., Thiery, J. P., Perez, E., and Pincet, F. (2005). Johnson-kendall-roberts theory applied to living cells. *Physical review letters*, 94(2):028102.

Däschinger, M., Knote, A., and von Mammen, S. (in press 2017). An evolutionary approach to behavioural morphometrics. In *GECCO '17 Companion*, Berlin, Germany. ACM.

Disset, J., Cussat-Blanc, S., and Duthen, Y. (2014). Self-organization of symbiotic multicellular structures. In *the Fourteenth International Conference on the Synthesis and Simulation of Living Systems-ALIFE 2014*, pages pp–541.

Drasdo, D., Hoehme, S., and Block, M. (2007). On the role of physics in the growth and pattern formation of multi-cellular systems: what can we learn from individual-cell based models? *Journal of Statistical Physics*, 128(1-2):287–345.

Goldberg, D. E. and Holland, J. H. (1988). Genetic algorithms and machine learning. *Machine learning*, 3(2):95–99.

Gros, C. (2014). Generating functionals for guided self-organization. In *Guided Self-Organization: Inception*, pages 53–66. Springer.

Hamant, O., Heisler, M. G., Jönsson, H., Krupinski, P., Uyttewaal, M., Bokov, P., Corson, F., Sahlin, P., Boudaoud, A., Meyerowitz, E. M., et al. (2008). Developmental patterning by mechanical signals in arabidopsis. *science*, 322(5908):1650–1655.

Hu, D. and Marcucio, R. (2009). A SHH-responsive signaling center in the forebrain regulates craniofacial morphogenesis via the facial ectoderm. *Development*, 136(1):107.

Hu, D., Marcucio, R., and Helms, J. (2003). A zone of frontonasal ectoderm regulates patterning and growth in the face. *Development*, 130(9):1749.

Jeong, W.-K., Beyer, J., Hadwiger, M., Blue, R., Law, C., Vázquez-Reina, A., Reid, R. C., Lichtman, J., and Pfister, H. (2010). Ssecrett and neurotrace: interactive visualization and analysis tools for large-scale neuroscience data sets. *Computer Graphics and Applications, IEEE*, 30(3):58–70.

Lobo, D. and Levin, M. (2015). Inferring regulatory networks from experimental morphological phenotypes: a computational method reverse-engineers planarian regeneration. *PLoS Comput Biol*, 11(6):e1004295.

Macklin, M., Müller, M., Chentanez, N., and Kim, T.-Y. (2014). Unified particle physics for real-time applications. *ACM Trans. Graph.*, 33:153:1–153:12.

Napp, N. and Nagpal, R. (2014). Distributed amorphous ramp construction in unstructured environments. *Robotica*, 32(02):279–290.

Narayanan, S. and Kidambi, P. (2011). Interactive simulations: History, features, and trends. In *Human-in-the-Loop Simulations*, pages 1–13. Springer.

Prokopenko, M. (2014). *Guided self-organization: Inception*, volume eBook. Springer, Berlin / Heidelberg.

Prokopenko, M., Barnett, L., Harré, M., Lizier, J. T., Obst, O., and Wang, X. R. (2015). Fisher transfer entropy: quantifying the gain in transient sensitivity. In *Proc. R. Soc. A*, volume 471, page 20150610. The Royal Society.

Prokopenko, M., Polani, D., and Ay, N. (2014). On the cross-disciplinary nature of guided self-organisation. In *Guided Self-Organization: Inception*, pages 3–15. Springer.

Salazar-Ciudad, I., Jernvall, J., and Newman, S. (2003). Mechanisms of pattern formation in development and evolution. *Development*, 130(10):2027–2037.

Salge, C., Glackin, C., and Polani, D. (2014). Empowerment–an introduction. In *Guided Self-Organization: Inception*, pages 67–114. Springer.

Santos, E. E., Guo, D., Santos Jr, E., and Onesty, W. (2004). A multi-agent system environment for modelling cell and tissue biology. In *PDPTA*, pages 3–9.

Sayama, H. (2014). Guiding designs of self-organizing swarms: Interactive and automated approaches. In *Guided Self-Organization: Inception*, pages 365–387. Springer.

Schambach, S. J., Bag, S., Schilling, L., Groden, C., and Brockmann, M. A. (2010). Application of micro-ct in small animal imaging. *Methods*, 50(1):2–13.

Soleymani, T., Trianni, V., Bonani, M., Mondada, F., and Dorigo, M. (2015). Bio-inspired construction with mobile robots and compliant pockets. *Robotics and Autonomous Systems*, 74:340–350.

Steele, J. and Iliinsky, N. (2010). *Beautiful visualization*. O'Reilly Media, Inc.

Swat, M. H., Thomas, G. L., Belmonte, J. M., Shirinifard, A., Hmeljak, D., and Glazier, J. A. (2012). Multi-scale modeling of tissues using compucell3d. *Methods in cell biology*, 110:325.

Uyttewaal, M., Traas, J., and Hamant, O. (2010). Integrating physical stress, growth, and development. *Current opinion in plant biology*, 13(1):46–52.

von Mammen, S. and Däschinger, M. (2015). Time series evolution for integrating developmental processes. In *Proceedings of the European Conference on Artificial Life 2015 (ECAL)*, pages 320–324.

von Mammen, S., Phillips, D., Davison, T., Jamniczky, H., Hallgrímsson, B., and Jacob, C. (2012). *Swarm-based Computational Development*, chapter 18, pages 473–499. Understanding Complex Systems. Springer Verlag.

Werfel, J., Petersen, K., and Nagpal, R. (2014). Designing collective behavior in a termite-inspired robot construction team. *Science*, 343(6172):754–758.

Xu, Q., Jamniczky, H., Hu, D., Green, R. M., Marcucio, R. S., Hallgrimsson, B., and Mio, W. (2015). Correlations between the morphology of sonic hedgehog expression domains and embryonic craniofacial shape. *Evolutionary Biology*, pages 1–8.

Zelditch, M. L., Swiderski, D. L., and Sheets, H. D. (2012). *Geometric morphometrics for biologists: a primer*. Academic Press.