

# Artificial Morphogenesis as an Example of Embodied Computation

BRUCE J. MACLENNAN

*Department of Electrical Engineering and Computer Science,  
University of Tennessee, Knoxville  
<http://www.cs.utk.edu/~mclennan>*

*Received: July 22, 2010. Accepted: November 8, 2010.*

*Embodied computation* is computation in which information processing emerges from and directly governs physical processes. As an example we present *artificial morphogenesis*, which uses computational processes analogous to those in embryological development in order to assemble complex physical structures. We discuss the requirements for a formalism or programming language for embodied computation oriented toward artificial morphogenesis and present a preliminary design for such a formalism. Use of the formalism is illustrated by several embodied computation problems of increasing complexity.

*Keywords:* algorithmic assembly, embodied computation, embodiment, embryological development, metamorphosis, morphogenesis, nanotechnology, post-Moore's Law computing, reconfigurable systems, self-assembly, self-organization

## 1 BACKGROUND

### 1.1 Embodied Computation

Pfeifer, Lungarella, and Iida [47] provide a concise definition of *embodiment*: “the interplay of information and physical processes.” Hence, *embodied computation* may be defined as computation whose physical realization is directly involved in the computational process or its goals [34, 39]. It includes computational processes that directly exploit physical processes for computational ends and those processes in which information representation and computation are implicit in the physics of the system and its environment, which effectively represent themselves. It also includes computational processes in which the intended effects of the computation include the growth,

assembly, development, transformation, reconfiguration, or disassembly of the physical system embodying the computation. Embodied computation is based on some of the insights from *embodied cognition* and *embodied artificial intelligence* [8, 10, 11, 16, 25, 26, 42, 46–48], but extends them to all computation [34, 39].

The most common model of computation (binary digital logic) is far removed from the physical processes by which it is implemented, and this has facilitated a beneficial independence of computer design from device technology. Thus our technological investment in computer design has been preserved through several generations of device technology (from relays to integrated circuits). We have had the luxury of using a large number of devices, operating sequentially, to implement each computational primitive (e.g., addition). This is because computation and the physical processes realizing it have existed at different scales of space and time. However, as in the coming decades we enter the era of post-Moore’s Law computing, increasing the density and speed of computation will require a greater assimilation between computational and physical processes [34–36, 39]. In part this will be accomplished by developing new physical systems and processes for computation, but the other half of the equation is to develop new models of computation that are closer to the laws of physics. The challenge is to identify models that are sufficiently low level to be readily implementable and sufficiently high level to be relatively independent of particular implementation technologies.

One of the advantages of embodied computing is that many computational processes are performed by physical systems “for free.” For example, neural networks often make use of universal approximation theorems based on linear combinations and simple sigmoidal functions [24, pp. 208–94]. However, sigmoidal behavior is a common consequence of many physical processes, in which some resource is exhausted, and therefore sigmoids are available “for free” without additional computation. Similarly, negative feedback often arises from the natural degradation or dissipation of physical substances, and so these processes are directly available for embodied computation. Further, stochastic effects, which are unavoidable in many physical systems, especially at the nanoscale, can be exploited as sources of *free variability* in algorithms such as stochastic resonance [4], simulated annealing [28], and other stochastic optimization processes.

Traditionally we have designed computers to operate with sequential logic, and then we have attempted to increase computation speed by using these sequential machines in parallel. In embodied computation, in contrast, the concurrency typical of physical, chemical, and biological processes is directly exploited to achieve parallelism. Sequentiality, where necessary, is imposed on an inherently parallel process. A simple example application of “parallelism for free” is diffusion, which occurs naturally in many fluids, such as liquids and gasses, and in other media; for example, cell-to-cell diffusion is critical in embryological morphogenesis [29]. Diffusion can be used for many

computational processes, including broadcasting information, optimization, constraint satisfaction, and massively parallel search, such as in path planning through mazes [27, 44, 49, 54, 58]. Diffusion is expensive to implement by conventional computation, but it comes for free in many physical systems.

A common tradeoff faced by many search algorithms is *exploration* versus *exploitation*, that is, the acquisition of new information versus the use of the information already obtained. Embodied computation systems often naturally and implicitly implement a dynamic balance between exploration and exploitation. A well known example is ant foraging behavior, in which ants imperfectly follow pheromone-marked trails to food sources [9]. Initially, random wandering implements unbiased exploration, but as knowledge is acquired, positive feedback biases activity toward exploitation. Built-in negative feedback (arising “for free” from degradation and dissipation of pheromones) ensures that in the absence of positive feedback, the balance shifts from exploitation back toward exploration. Thus simple physical processes implement a parallel control system that sensitively and robustly manages the acquisition and use of information.

*Cell-sorting by differential adhesion* is an example, from embryological development, of a natural embodied computation process that makes productive use of physical phenomena [17, ch. 4]. In this process there is a mixed population of cells with different degrees of cohesion. Under conditions of random motion (e.g., undirected wandering), the cells sort themselves out into spatially separated groups. In the absence of constraints, the cells form two concentric spheres, with the more tightly cohering particles in the center. In the presence of constraints, the particles sort themselves into separated tissues or bodies. This is an important process in embryological morphogenesis, and may be useful in artificial morphogenesis in nanotechnology and related applications. A similar process is *lumen formation* resulting from polarized cells with nonuniform distributions of adhesion molecules [17, pp. 78–80]. Under random motion the cells sort themselves into a low-energy configuration in which there are hollows (lumens) bounded by the less adhesive faces of the cells.

Nature also provides informative examples of how the physical system may be its own representation, which are relevant to the application of computational ideas in nanotechnology. For example, *stigmergy* refers to the process wherein the “project” undertaken by one or more organisms embodies the information required to continue and complete the project [9]. The best-known example is wasp nest building [7]. The partially completed nest itself provides the stimuli that guide the individual wasps in the construction process. Therefore there is no need for the wasps to have representations of the completed nest or of the current state of its construction, or to have an internal “program” for nest construction. In this way, relatively simple agents (with modest information processing capacity) can construct complex, functional structures.

Typically, the structure of a computational system governs its function, that is, the computation it performs. Conversely, in *algorithmic assembly* a computational process governs the assembly of some physical structure [3,30,31,50,51]. However, some embodied systems integrate these two processes. For example, in embryological morphogenesis, the physical structure of the embryo governs the very computational processes that create the physical structure of the embryo. Structure governs function, and function creates structure.

Further, since embodied computation systems are potentially capable of modifying their own structure, they can be naturally adaptive. Beyond this, they may be “radically reconfigurable,” that is, able to reorganize their physical structure to adapt to changing circumstances and objectives [33]. A related, but very important, property is self-repair, since acceptable configurations often can be defined as stationary states to which the system automatically reconfigures after damage. Finally, embodied computation systems can be designed for self-destruction, which is especially important for nanoscale systems. If they can *reconfigure* themselves, they can also *deconfigure* themselves, rearranging their components into inert and potentially recyclable material. As we know, apoptosis (programmed cell death) is essential both in embryological development and in the maintenance of a well-functioning body.

Despite the opportunities of embodied computing, there are also challenges. We are used to programming in an idealized world of perfect logic, independent of physical realization, and therefore embodied computing presents unfamiliar problems, for we have to pay more attention to the physical realization of an embodied computation and its environment. However, embodied computation is not simply applied physics, for although embodied computation makes more direct use of physical processes than does conventional computation, its focus is on processes that are relatively independent of specific physical systems, so that they can be applied in a variety of physical substrates. For example, reaction-diffusion systems can be instantiated in a wide variety of media [1]. Further, since natural computation is embodied, we often can look to natural systems to learn how to implement artificial embodied computing systems.

## 1.2 Artificial Morphogenesis

Our research is currently focused on a particular application of embodied computation that is of central importance in nanotechnology: artificial morphogenesis [35,38]. While a variety of self-assembly processes have been applied to nanomaterials, these have not solved the problem of assembling systems that are hierarchically structured from the nanoscale up to the macroscale. Some of the issues are being addressed in the context of research in programmable matter and reconfigurable robotics, but we believe that these efforts would be improved by a more morphogenetic approach, for

embryological morphogenesis is the best example that we have of successful self-assembly of physical systems structured from the nanoscale up to the macroscale.

Morphogenesis has a number of special characteristics that distinguish it from most other self-organizing processes, and we believe that these characteristics will be important in embodied computation. For example, we commonly think of computation as taking place in a fixed substrate, and many self-assembly processes also assume a fixed substrate or matrix in which agents move. In morphogenesis, in contrast, the computational medium is assembled by the computational process, as a zygote develops into a hollow blastula and then into a more complex structure of tissues, which govern the information and control processes in the medium. Although cellular automaton (CA) models, for example, have been applied productively to the study of localized pattern formation processes, they are inadequate for describing morphogenesis as a whole. CA models assume a predefined regular spatial grid, whereas biological morphogenesis creates the space (the embryo) in which (and relative to which) development occurs. Generally speaking, in nature self-organization proceeds without the benefit of fixed, predefined reference frames and coordinate systems, which is one source of the robustness of these processes.

In morphogenesis, *tissues* (groups of cells with a common function) form and reform under control of their inherent self-organizing processes. We think of the embryo as solid, but most of the tissues are elastic, at least during development, and elastic properties influence the forms that develop [56, ch. 6]. In other cases, tissues behave more like viscous fluids, perhaps percolating through a more solid matrix, and this fluid motion is essential to cell migration during development [5][17, pp. 92–4][55]. Non-cellular substances, such as morphogens and other signaling chemicals diffuse like gases through anisotropic media, but cells also exhibit facilitated diffusion [17, pp. 13–15, 156, 252]. In many cases, tissues occupy a middle ground between solid and fluid, with viscoelastic properties [17, pp. 21–2, 133]. Generally speaking, morphogenesis takes place in the relatively unexplored realm of *soft matter* [13][17, p. 2], and our theories of embodied computation, at least as applied to morphogenesis, need to take account of its characteristics.

Morphogenesis proceeds through a carefully orchestrated series of overlapping parallel phases, which have the characteristics of a *coordinated algorithm* [57]. In this robust process, the completion of one phase signals the initiation of the next through a combination of chemical signals and changing cell states. Temporal patterns often create spatial patterns (as in the clock-and-wavefront model of segmentation [12, 15, 22]), and morphogenesis may be best understood as the creation of patterns in *four* dimensions [21, p. 504n].

Developmental biologists have identified about twenty fundamental processes involved in morphogenesis [17, pp. 158–9][52]. If indeed, these processes are sufficient to produce complex, hierarchically-structured systems,

such as vertebrate organisms, then they define an agenda for embodied computation applied to artificial morphogenesis.

## 2 REQUIREMENTS FOR A FORMALISM

One goal of our research is to develop and evaluate formal methods for embodied computation oriented toward artificial morphogenesis. This implies several requirements.

First, the goals of embodied computation are best served by a continuous perspective, since the laws of physics are primarily continuous (ordinary and partial differential equations). This is especially true in our intended application area, morphogenesis, for tissues and their environments are naturally treated as continua (e.g., epithelia, mesenchyme, blood). Furthermore, the objective of post-Moore's Law computing is greater densities, greater speed, and greater parallelism, all of which make computation, at a macroscopic scale, look like a continuous process occurring in a continuous medium. While some of these phenomena are *physical continua* (e.g., electrical fields), others are *phenomenological continua* composed of microscopic discrete *units* (atoms, molecules, cells, microrobots, etc.).

Nevertheless, the two perspectives — the discrete and the continuous — are complementary. In many applications of embodied computation, especially when the computational process is implemented by very large numbers of computational units, we will want to be able to move fluently between the two perspectives. This is especially the case in morphogenesis, where we are faced with both discrete and continuous phenomena. In the early stages of development there is a small discrete number of cells (1, 2, 4, 8, etc.) with specific shapes and arrangements; in later stages, when there are more than  $\sim 10\,000$  cells, it is more convenient to treat the cell masses as viscoelastic tissues and apply continuum mechanics [17]. Therefore a formalism for embodied computation should support a systematic ambiguity between discrete and continuous models. That is, so far as possible, the formalism should be interpretable as describing a mathematical continuum or a large set of discrete units, and as describing either a continuous- or discrete-time process.

Thus, the formalism should support complementarity by treating bodies, tissues, and other macroscopic masses as comprising an indefinitely large number of *elements*, which we interpret ambiguously as infinitesimal points in a continuum or as members of a finite, discrete set of units. Adapting the terminology of continuum mechanics, we call these elements *particles* when it is more convenient to think of them as discrete units, and *material points* when it is more convenient to think of them as infinitesimal points in a continuum. The continuous perspective also helps ensure that algorithmic processes scale to very large numbers of units.

To maintain these complementary interpretations, the formalism should treat the elements as having a small, indeterminate size, and therefore the formalism should describe the properties of elements in terms of *intensive quantities*, such as number density and mass density, which do not depend on size, in preference to *extensive quantities*, such as volume and mass, which do. The individual cells constituting a tissue may have definite shapes, orientations, etc. that are relevant to the morphogenetic process (e.g., controlling cell adhesion, cell migration, etc.), and so these properties cannot be ignored. The solution is to treat these properties as intensive quantities (e.g., aspect ratios, probability density functions of orientations).

The formalism should be suitable for describing embodied computing processes in all kinds of media, including solids, liquids, gases, and physical fields. Further, viscoelastic media (so-called “soft matter” [13]) are important in morphogenesis; for example, mesenchyme is best characterized as viscoelastic [17, p. 98][18]. Therefore, we expect to be dealing with materials of differing viscosity and elasticity. Indeed, soft matter might be a new metaphor in morphogenesis, succeeding the crystal, field, and fabric metaphors discussed by Haraway [23]. Finally, some of these materials may be anisotropic (e.g., epithelium), and our formalism must accommodate that possibility.

The formalism must be capable of describing active elements, such as living cells and microrobots, as well as passive elements, such as diffusing chemicals, fluid media, and solid substrates. In particular, the formalism should be applicable to living agents as well as to nonliving ones, for embryological morphogenesis, which is the inspiration for this technology, is based on living cells, and we also want to address artificial morphogenetic processes based on genetically engineered organisms and other products of synthetic biology.

Energetic issues are critical to embodied computational systems, which must be maintained in a nonequilibrium state either for a definite duration or indefinitely. Therefore the formalism needs to be able to describe and define the flow of matter and energy through the system. Active elements have to be powered in some way, either continuously during operation or by being arranged initially in a nonequilibrium state. We anticipate that embodied computational systems will be fueled in a variety of ways and powered by a variety of energy sources (electrical, chemical, optical, thermal, mechanical, etc.). Therefore the formalism needs to be able to define the dynamical and spatial relations among energy sources and other elements. Dissipation of energy is especially critical for microscopic nonequilibrium systems, and should be addressed by the formalism. Similarly, embodied computation must address the disposal of other waste products, such as unrecyclable chemical reaction products. The elimination of waste products may be implemented by diffusion, convection, or more structured transport processes, as defined in the embodied computation system.

Descriptions of morphogenetic and developmental processes in terms of partial differential equations are frequently expressed relative to a fixed, external three-dimensional reference frame. There are two problems with this. First, the natural reference frame is the developing body itself, which might not have any significant relationship to a fixed frame (e.g., the developing embryo, with its intrinsic frame, in variable relation to a terrestrial frame). Second, since active elements (such as migrating cells) are responding to their local environments (e.g., chemical gradients in their immediate vicinity), it is natural to describe their behavior in terms of their intrinsic coordinates (e.g., ahead/behind, above/below) or their immediately local frame (e.g., the substrate on which or medium through which they are migrating). Therefore tensors seem to be the best way to describe the properties and behaviors of elements.

It is essential that a formalism for embodied computation integrate smoothly with the usual mathematical formulations of physical, chemical, and biological processes. It should be easy to translate between computational and mathematical expressions so that the full range of physical theory and mathematical method can be applied to embodied computation. On the one hand, this will permit mathematical analysis to be applied, when the appropriate methods and results exist. On the other hand, since many of these systems are complex and nonlinear, the power of contemporary mathematical analysis may be insufficient to understand these systems. Therefore, the formalism should facilitate simulation of a system by straightforward translation into behavioral rules (“programs”) for the elements. The formalism should be operational as well as analytical, that is, a programming language.

Due to the mathematical intractability of many interesting systems, we expect embodied computation to have a significant experimental component. Thus, as already mentioned, we expect simulation to play an important role in the design, evaluation, and testing of embodied computation systems before they are physically realized. In most respects the required simulation tools will be similar to those commonly used in computational science, but there are at least two differences.

First, biologists often find it convenient to express regulatory networks qualitatively, using *influence models* to indicate, for example, that one gene product enhances or represses the expression of another gene, or that a molecule regulates an enzyme [59]. These qualitative regulatory relationships are an important tool for conceptualizing control processes in which the quantitative relationships are unknown or are not considered critical. Therefore, the formalism should permit the expression of qualitative control relationships.

Second, in embodied computation, especially as it pushes towards the nanoscale, noise, error, uncertainty, defects, faults, and other sources of indeterminacy and unpredictability are unavoidable [32]. They should not be treated as secondary effects, taken into account only as an afterthought, but rather as essential properties of the medium of embodied computation.

Thus we seek first to exploit them as computational resources — sources of *free variability* — and only if that fails, to minimize them as interference. Therefore a formalism for embodied computation should be oriented toward the description of stochastic processes. Further, since many self-organizing processes depend on stochasticity and sampling effects for symmetry breaking, the formalism should admit them (in effect as processes in which discrete phenomena can influence structures in continuous media).

### 3 A FORMALISM FOR EMBODIED COMPUTING

Having outlined the requirements for a formalism for embodied computation, we present a preliminary description.

**Substances:** One of the central concepts in the proposed formalism is a *substance*, which refers to a class of phenomenological continua with similar properties, especially from a computational perspective (e.g., cohesion, viscosity, behavioral repertoire). They are defined by (perhaps fuzzy) ranges of parameter values, ratios of parameters, etc. Some of the most familiar substances are solids, liquids, and gases. Other useful classes are incompressible, viscous, elastic, and viscoelastic substances. Our notion of substance includes other spatially distributed quantities, such as electromagnetic, optical, thermal, and gravitational fields (more abstractly, scalar, vector, and tensor fields).

Substances are naturally organized in a hierarchy, from the most general classes (e.g., solid, liquid, gas) to specific physical substances (e.g., liquid water, oxygen gas, mesenchyme, a specific extracellular matrix). (Of course there are borderline cases.) The more generic classes are more useful from a computational perspective, since they have the potential of being realized by a greater variety of specific substances. For example, for the purposes of embodied computation it may be sufficient that a substance diffuse and degrade at certain relative rates, and be producible and detectable by other particles, but the choice of specific substance might be otherwise unconstrained.

It is apparent that substance hierarchies have similarities to class hierarchies in object-oriented programming, but there are also some differences. First, as will be explained below, the instances of substance classes are continuous *bodies* (or *tissues*) rather than discrete atomic objects, which are the instances of classes in object-oriented programming. Second, while substances are similar to classes in that they are defined by subclass relationships, by variables associated with their particles, and by common behaviors, in addition substances are defined by constraints on the values of these variables. Thus, substance definitions are more like definitions in mathematics or physics than like class definitions in an object-oriented programming language. Nevertheless, it is our intent that substance definitions be sufficiently formal that they can be used in simulations (i.e., as programs).

**Bodies:** In object-oriented programming the instances (members) of classes are discrete atomic objects, which frequently act as software agents. In our formalism for embodied computing, in contrast, instances of substances are called *bodies* or *tissues*, which are phenomenological continua of elements (discrete *particles* or infinitesimal *material points*). Typically bodies are homogeneous, that is, composed of elements of a particular type that is characteristic of the substance's definition (e.g., water molecules, cells in a particular state of differentiation).

Several bodies may occupy the same region of space and interact with each other. For example, the same region may be occupied by a volume of diffusing chemical and by a mass of cells following the chemical gradient. The kind and degree of interpenetrability possible, as well as other interactions, are determined by the definitions of the substances. Some bodies may be quite diffuse (e.g., disconnected cells migrating through mesenchyme).

Mathematically, a body is defined to be an indefinitely large set  $\mathcal{B}$  of elements (particles or material points). We say "indefinitely large" to maintain the systematic ambiguity between the infinite number of material points in a continuum and the finite but large number of particles in a discrete ensemble. It is often convenient to think of the elements of  $\mathcal{B}$  as *labels* or *identifiers* for the elements. At any given time  $t$  each element  $P \in \mathcal{B}$  has a location in a region of three-dimensional Euclidean space  $\mathcal{E}$  defined by a vector  $\mathbf{p} = C_t(P)$ , where  $C_t$  is a continuous function that maps  $\mathcal{B}$  into this region. As in continuum mechanics,  $C_t$  defines the *configuration* of  $\mathcal{B}$  at time  $t$ , and therefore  $C_t$  reflects the *deformation* of the body as a consequence of its own internal dynamics and its interaction with other bodies.

An embodied computation system comprises a finite and fixed number of bodies (or tissues), each having properties that allow it to be classified as one substance or another. The behavioral dynamics of these bodies, in mutual interaction, defines the dynamics of the embodied computation system, but it must be prepared in some appropriate initial state. This is accomplished by specifying that a particular body of a defined substance occupies a specified region of Euclidean space and by specifying particular initial values for the variable properties of the substance throughout that region (i.e., for each element constituting the body). (In many cases the values will be uniform throughout the region or vary randomly according to some distribution.) Examples include a concentration of a substance in a small volume (e.g., an injected chemical or an inoculation of identical cells), the definition of a uniform gravitational or electrical field, and a bath at a specified temperature and occupying a defined region. While the full generality of mathematics may be used to define the initial bodies, we are most interested in physically feasible preparations, but this depends on the specifics of the embodied computation system.

**Elements:** A substance is defined in terms of the properties and behaviors of its elements, and so we need to consider how they may be defined

consistently with the requirements of Sec. 2. For most purposes the formalism makes use of *material* (*Lagrangian, reference*) descriptions of these properties and behaviors, rather than a *spatial* (*Eulerian*) reference frame. This means that we consider a physical quantity  $Q$  as a time-varying function of a fixed particle  $P \in \mathcal{B}$  as it moves through space,  $Q(P, t)$ , rather than as a time-varying property  $q(\mathbf{p}, t)$  at a particular fixed location in space,  $\mathbf{p} \in \mathcal{E}$ . For example, if temperature is the property, then we think of the temperature as a (perhaps variable) property of a moving particle, as opposed to thinking of a (time varying) temperature field with respect to a fixed spatial reference frame. In effect, the use of material variables in preference to spatial variables is a particle-centered description, and is a more object-oriented or agent-based way of thinking in that we can think of the particles as carrying their own properties with them and behaving like well-defined entities. (Obviously the two reference frames are related by the configuration function,  $C_t$ .) The distinction is illustrated by velocity. At any given time each particle  $P$  has a velocity  $\mathbf{V}$ , which is the material time derivative of its (coordinate-independent) position vector,  $\mathbf{p}$ :  $\mathbf{V}(P) = D_t \mathbf{p} = D_t C_t(P)$ . On the other hand, the spatial derivative  $\mathbf{v}(\mathbf{p}) = \mathbf{V}[C_t^{-1}(\mathbf{p})]$  defines a velocity vector field with respect to the spatial frame (the rate of change of spatial position at each spatial position).

As explained in Sec. 2, in order to maintain independence of the size of the elements, so far as possible all quantities are *intensive*. This is one of the characteristics that distinguishes this model of embodied computing from ordinary mathematical descriptions of physical phenomena. For example, instead of elements having a mass (which is an extensive quantity), they have a (mass) density, the mass per unit volume, which is an intensive quantity and doesn't depend on the size of the elements. Similarly, instead of dealing with  $N$ , the number of particles (e.g., cells or molecules) corresponding to an element, we deal with its *number density*  $n$ , the number of particles per unit volume. Note that if the number density becomes very small, then the dynamics will be subject to small sample effects, which are often important in self-organization; they are discussed further below. There are various indicators of when continuous mathematics is a good approximation. For example, the *Knudsen number*  $\text{Kn} = \lambda/L$  is the ratio of a particle's mean free path length  $\lambda$  to a characteristic length scale  $L$ , such as its diameter. In the case of morphogenesis, this could measure how many diameters a cell is likely to move before its internal control processes can change its direction. For low Knudsen numbers ( $< 1$ ) it is safe to apply the approximations of continuum mechanics, but high numbers may require the methods of statistical mechanics.

The requirement for complementary perspectives implies that the properties of elements be treated as bulk quantities, that is, as the collective properties of an indeterminate ensemble of *units* (e.g., cells or molecules). This creates special requirements when the units constituting an element might have differing values for an attribute. For example, cells, molecules, and microrobots

have an orientation, which is often critical to their collective behavior. In some cases all the units will have the same orientation, in which case the orientation can be treated as an ordinary intensive property of the element. In most cases, however, we must allow for the fact that the units may have differing orientations (even if a consequence only of defects or thermal agitation). Therefore, we have to consider the *distribution* of orientations; each orientation (represented by a unit vector  $\mathbf{u}$ ) has a corresponding probability density  $\Pr\{\mathbf{u}\}$ . If  $n$  is the number density of an element, then  $n \Pr\{\mathbf{u}\}$  is the number density of units with orientation  $\mathbf{u}$ . Equivalently, the orientation may be interpreted as a vector-valued random variable,  $\mathbf{U}$ .

Morphogenesis in development sometimes depends on cell shape; for example, a change from a cylindrical shape to a truncated cone may cause a flat tissue to fold [17, pp. 113–16][45]. Generally a shape can be expressed as a vector or matrix of essential parameters that define relevant aspects of the shape. Two complexities arise in the expression of shape in this formalism for embodied computing. First, the shape must be expressed in a coordinate-independent way, which means that we are dealing with a *shape tensor*. Second, since an element represents an indefinite ensemble of units, shape must be treated as a probability distribution defined over shape tensors, or as a tensor-valued random variable.

One of the advantages of object-oriented programming and agent-based simulations is the ability to describe and reason about each agent as an individual, obeying its own behavioral rules [6]. It may seem that the proposed formalism loses this attractive feature because, while the individual units may behave like independent agents (and have, for example, a definite orientation), the units are below the level at which the formalism operates. Rather, it operates at the level of elements, which comprise an indefinite number of units (each of which may have its own orientation). However, we can recover some of the intuitive understandability of agent-based descriptions by thinking of the elements as quasi-agents with indefinite properties. For example, we can think of an element as having an indeterminate orientation, with the probability of a specific orientation given by the distribution of orientations in that element.

Therefore, we treat the properties of elements as random variables with associated probability distributions. The distributions are determined by the values of the constituent units, but that is below the level of abstraction of the formalism, for which the distributions are taken as primitive. This does not solve all the problems, however, and in particular we must take care of small sample effects, which may be crucial to self-organization. Another issue is how to manipulate reliably non-independent random variables. Therefore one goal of our project is to formulate rules of inference that allow reliable description and control of elements as quasi-agents.

Morphogenetic processes, both natural and artificial, can be described at many different levels, from individual units (cells, microrobots, molecules) to

masses of units that can be treated as infinitesimal elements of a continuum. The latter is our approach, since it admits of the widest range of possible physical realizations and because it focuses on morphogenetic algorithms that will scale up to very large numbers of units. To establish that a proposed agent-based implementation is correct, one must show that the continuous equations are a sufficiently accurate description of the behavior of the agents en masse. To mention a specific example, a stochastic differential equation describing the random motion of an individual unit may realize a Fokker-Planck equation defining the change in number density of the units (see Sec. 4.1).

To summarize the preceding discussion, all the variables representing the state of an element should, so far as possible, be intensive quantities, coordinate-independent (i.e., tensors), and generally interpreted as random variables with an associated probability distribution.

**Behavior:** The behavior of the elements (particles, material points) of a body is defined by rules that describe the temporal change of various quantities, primarily intensive tensor quantities (coordinate-independent scalars, vectors, and higher-dimensional objects). Such changes might be expressed in continuous time, by ordinary differential equations, e.g.,  $D_t X = F(X, Y)$ , or in discrete time by finite difference equations, e.g.,  $\Delta_t X = F(X, Y)$ , where  $\Delta_t X = \Delta X / \Delta t$ ,  $\Delta X = X(t + \Delta t) - X(t)$ , and the time increment  $\Delta t$  is implicit in the  $\Delta_t$  notation. Generally, these would be *substantial* or *material* (as opposed to *spatial*) time derivatives, that is, derivatives evaluated with respect to a fixed particle:  $D_t X = \partial X(P, t) / \partial t|_{P \text{ fixed}}$ . Similarly, we use material differences.

In order to maintain complementarity between discrete and continuous time, the proposed formalism expresses such relationships in a neutral notation:

$$\mathfrak{D}X = F(X, Y).$$

This *change equation* may be read, “the change in  $X$  is given by  $F(X, Y)$ .” The rules of manipulation for the  $\mathfrak{D}$  operator respect its complementary discrete- and continuous-time interpretations.

A particle-oriented description of behavior implies that in most cases active particles (e.g., cells, microrobots) will not have direct control over their position or velocity. Rather, particles will act by controlling local forces (e.g., adhesion, stress) between themselves and other particles in the same body or in other bodies. Therefore, “programming” active substances will involve implementing change equations that govern stress tensors and other motive forces associated with the particles.

It not so difficult to program change equations as might be supposed. Quantities that quickly saturate can be used like Boolean values, and Heaviside (unit step) functions can be used multiplicatively to turn processes on or off depending on conditions, such as whether a quantity is greater than

a threshold. In this way quite complex systems of hierarchical control can be implemented in terms of continuous variables. (Examples can be found in a technical report [37].) This may seem to be an indirect process, effectively using analog computation to implement digital control, but it is a more accurate reflection of conditions at the nanoscale, where digital perfection is unrealistic.

As discussed in Sec. 2, noise, uncertainty, error, faults, defects, and other sources of randomness are unavoidable in embodied computation, especially when applied in nanotechnology. Indeed, such randomness can often be exploited as a computational resource — *free variability* — in embodied computing. Therefore, the behavior of elements will often be described by stochastic differential/difference equations. To facilitate the complementary continuous/discrete interpretations, it is most convenient to express these equations in the Langevin form:

$$\mathfrak{D}X = F(X, Y, \dots) + G_1(X, Y, \dots)v_1(t) + \dots + G_n(X, Y, \dots)v_n(t),$$

where the  $v_j$  are noise terms, but we must be careful about the interpretation of equations of this kind.

To see why, consider a stochastic integral,  $X_t = \int_0^t H_t dW_t$ , where  $W$  is a Wiener process (Brownian motion). In differential form this is  $dX_t = H_t dW_t$ . To maintain complementarity, this should be consistent in the limit with the difference equation:  $\Delta X_t = H_t \Delta W_t$ . This implies that the stochastic integral is interpreted in accord with the Itô calculus. The corresponding stochastic change equation in Langevin form is  $\mathfrak{D}X_t = H_t \mathfrak{D}W_t$ . Interpreted as a finite difference equation it is  $\Delta_t X_t = H_t \Delta_t W_t$ , which makes sense. However the corresponding differential equation,  $D_t X_t = H_t D_t W_t$ , is problematic, since a Wiener process is nowhere differentiable. Fortunately we can treat  $D_t W_t$  purely formally, as follows. First observe that  $\Delta W_t = W_{t+\Delta t} - W_t$  is normally distributed with zero mean and variance  $\Delta t$ ,  $\mathcal{N}(0, \Delta t)$ . Therefore  $\Delta_t W_t = \Delta W_t / \Delta t$  is normally distributed with unit variance,  $\mathcal{N}(0, 1)$ , and  $\Delta_t W_t$  can be treated as a random variable with this distribution. To extend this to the continuous case, we treat  $\mathfrak{D}W_t$  as a random variable, distributed  $\mathcal{N}(0, 1)$ . Therefore the stochastic change equation  $\mathfrak{D}X = H \mathfrak{D}W$  has consistent complementary interpretations. Similarly, for an  $n$ -dimensional Wiener process  $\mathbf{W}^n$ , we interpret  $\mathfrak{D}\mathbf{W}^n$  to be an  $n$ -dimensional random vector distributed  $\mathcal{N}(0, 1)$  in each dimension.

As explained above (Sec. 2), one requirement for the formalism is that it be able to express qualitative behavioral rules corresponding to the influence diagrams widely used in biology. Therefore, we define the notation  $\mathfrak{D}X \sim -X, Y, Z$  (for example) to mean that the change of  $X$  is “repressed” (negatively regulated) by  $X$  and “promoted” (positively regulated) by  $Y$  and  $Z$ . We have been calling such a relationship a *regulation* and read it, “the change in  $X$  is negatively regulated by  $X$  and positively regulated by  $Y$  and  $Z$ .” Formally,

$\mathbb{D}X \sim -X, Y, Z$  is interpreted as a change equation  $\mathbb{D}X = F(-X, Y, Z)$  in which  $F$  is a function that is unspecified except that it is monotonically non-decreasing in each of its arguments, so that the signs of the arguments express positive or negative regulation.

## 4 EXAMPLES

### 4.1 Diffusion

Diffusion is an Itô process in an  $m$ -dimensional diffusion medium:

$$\mathbb{D}\mathbf{p} = \boldsymbol{\mu} + \boldsymbol{\sigma} \mathbb{D}\mathbf{W}^n,$$

where  $\boldsymbol{\mu}$  is a vector field in  $\mathbb{R}^m$  characterizing drift,  $\boldsymbol{\sigma}$  is an  $m \times n$  tensor field characterizing diffusibility in various directions and locations, and  $\mathbf{W}^n$  is an  $n$ -dimensional Wiener process driving the stochastic behavior. The drift velocity, which represents directed movement, may result from external forces (e.g., gravity, electrical field) or from particle activity (e.g., chemotaxis). The diffusion tensor, which represents constrained undirected movement, may result from Brownian motion or from active wandering [17, pp. 14–15].

A higher level description is given by the Fokker-Planck equation, which describes the time-varying probability density function of particles (a scalar field defined over the diffusion medium):

$$\mathbb{D}\phi = \nabla \cdot [-\boldsymbol{\mu}\phi + \nabla \cdot (\boldsymbol{\sigma}\boldsymbol{\sigma}^T \phi)/2].$$

Hence,  $\boldsymbol{\mu}$  represents the average particle velocity, and the rank-2 tensor  $\boldsymbol{\sigma}\boldsymbol{\sigma}^T/2 \in \mathbb{R}^{m \times m}$  represents the diffusion rates in various directions.

To give an idea of how simple diffusion could be expressed in operational terms, we use a programming-language-like notation:

**substance morphogen:**

<b>scalar field</b> $\phi$	concentration
<b>vector fields:</b>	
$\mathbf{j}$	flux
$\boldsymbol{\mu}$	drift vector
<b>order-2 field</b> $\boldsymbol{\sigma}$	diffusion tensor

**behavior:**

$\mathbf{j}$	$=$	$\boldsymbol{\mu}\phi - \nabla \cdot (\boldsymbol{\sigma}\boldsymbol{\sigma}^T \phi)/2$	flux from drift and diffusion
$\mathbb{D}\phi$	$=$	$-\nabla \cdot \mathbf{j}$	change in concentration

To describe the body constituting the system and its initial preparation we use a notation such as the following, which describes a small spherical concentration of the morphogen in the center of a cylindrical volume of the medium:

**body** Concentration of morphogen:  
**for**  $X^2 + Y^2 \leq 1$  **and**  $-1 \leq Z \leq 1$ :  
 $\mathbf{j} = 0$       || initial 0 flux  
 $\boldsymbol{\mu} = 0$       || drift vector  
 $\boldsymbol{\sigma} = 0.1\mathbf{I}$     || isotropic diffusion  
**for**  $X^2 + Y^2 + Z^2 \leq 0.1$ :  $\phi = 1$     || initial density inside sphere  
**for**  $X^2 + Y^2 + Z^2 > 0.1$ :  $\phi = 0$     || zero density outside sphere

#### 4.2 Activator-Inhibitor System

Activator-inhibitor systems are two-component reaction-diffusion systems that have proved useful as models of pattern formation in development [20, 40, 41, 60]. Both components diffuse, the activator  $A$  promoting the production of both, the inhibitor  $I$  repressing production. In the notation of the formalism, this is:

**substance** activator-inhibitor:  
**scalar fields:**  
 $A$               || activator concentration  
 $I$                 || inhibitor concentration  
**order-2 fields:**  
 $\boldsymbol{\sigma}_A$           || activator diffusion tensor  
 $\boldsymbol{\sigma}_I$           || inhibitor diffusion tensor  
**behavior:**  
 $\mathcal{D}A \sim A, -I, \Delta(\boldsymbol{\sigma}_A \boldsymbol{\sigma}_A^T A)$   
 $\mathcal{D}I \sim A, -I, \Delta(\boldsymbol{\sigma}_I \boldsymbol{\sigma}_I^T I)$

Here  $\Delta = \nabla \cdot \nabla$  is the Laplacian operator on tensor fields.

#### 4.3 Deneubourg Model

Deneubourg's model of pillar construction in wasp nests provides a good example of a continuous model of a discrete process [14]. Termites  $T$  deposit pheromone-bearing cement  $C$  at rate  $k_1$ , from which the pheromone evaporates at rate  $k_2$ :

**substance** cement:  
**scalar field**  $C$               || pheromone-bearing cement conc.  
**scalars:**

$$\begin{array}{ll} k_1 & \parallel \text{deposition rate} \\ k_2 & \parallel \text{pheromone evaporation rate} \end{array}$$

**behavior:**

$$\mathcal{D}C = k_1 T - k_2 C \quad \parallel \text{deposition} - \text{evaporation}$$

In this case we use change equations rather than regulations in order to express the conserved relations implied by the shared constants  $k_1$  and  $k_2$ . The air-born pheromone  $\phi$  degrades at rate  $k_3$  and diffuses subject to  $\mathbf{D}_\phi$ :

**substance pheromone is cement with:**

$$\begin{array}{ll} \text{scalar field } \phi & \parallel \text{concentration in air} \\ \text{scalar } k_3 & \parallel \text{degradation rate} \\ \text{order-2 field } \mathbf{D}_\phi & \parallel \text{diffusion tensor field} \end{array}$$

**behavior:**

$$\mathcal{D}\phi = k_2 C - k_3 \phi + \Delta(\mathbf{D}_\phi \phi) \quad \parallel \text{evap. from cement} - \text{degr.} + \text{diff.}$$

The concentration  $T$  of cement-laden termites is given by:

**substance termite-mass is pheromone with:**

$$\begin{array}{ll} \text{scalar field } T & \parallel \text{density} \\ \text{vector field } \mathbf{v} & \parallel \text{velocity field} \\ \text{scalar } r & \parallel \text{input flux of cement-laden termites} \\ \text{order-2 field } \mathbf{D}_T & \parallel \text{diffusion (wandering) tensor field} \\ \text{scalar } k_4 & \parallel \text{strength of gradient following} \end{array}$$

**behavior:**

$$\begin{array}{ll} \mathbf{v} = k_4 \nabla \phi - T^{-1} \nabla \cdot \mathbf{D}_T T & \parallel \text{gradient following} - \text{diffusion} \\ \mathcal{D}\mathbf{p} = \mathbf{v} & \parallel \text{change in position} \\ \mathcal{D}T = r - k_1 T - \nabla \cdot (T\mathbf{v}) + \mathbf{v} \cdot \nabla T & \parallel \text{change in material frame} \end{array}$$

Recall that  $\mathbf{p}$  (spatial position) is a property of all substances, and so the equation for  $\mathcal{D}\mathbf{p}$  defines the motion (displacement and deformation) of the termite mass. In general, when a particle  $P$  in one body refers to a variable  $Q$  characteristic of another body, the variable is evaluated at the same spatial location as  $P$ . For example, these equations for *termite-mass* refer to  $\phi$ , which is a property of the substance *pheromone*. Furthermore, gradients such as  $\nabla \phi$  are evaluated in the spatial (Eulerian) frame, since this reflects the configuration of the body at a given time.

#### 4.4 Vasculogenesis

For our next example we extend the model developed by Frederico Bussolino and his colleagues of vasculogenesis, the early stages of formation of capillary networks from dispersed endothelial cells [2, 19, 53] (cf. [43]). Aggregation is governed by a morphogen that is released by the cells and that diffuses and degrades.

**substance** morphogen:

**scalar fields:**

$C$  || concentration

$S$  || source

**order-2 field**  $\mathbf{D}$  || diffusion tensor

**scalar**  $\tau$  || degradation time constant

**behavior:**

$$\mathbb{D}C = \Delta(\mathbf{D}C) + S - C/\tau \quad \text{|| diffusion + release - degradation}$$

The cells produce morphogen at a rate  $\alpha$  and follow the gradient at a rate governed by attraction strength  $\beta$ . Cell motion is impeded by dissipative interactions with the substrate, which are measured by an order-2 tensor field  $\gamma$ . Since cells are filled with water, they are nearly incompressible, and so they have a maximum density  $n_0$ , which influences cell motion; to accommodate this property (which will also apply to microrobots), the model uses an arbitrary function,  $\phi$ , that increases very rapidly as the density exceeds  $n_0$ .

**substance** cell-mass is morphogen **with:**

**scalar fields:**

$n$  || number density of cell mass

$\phi$  || cell compression force

**vector field**  $\mathbf{v}$  || cell velocity

**scalars:**

$n_0$  || maximum cell density

$\alpha$  || rate of morphogen release

$\beta$  || strength of morphogen attraction

**order-2 field**  $\gamma$  || dissipative interaction

**behavior:**

$$S = \alpha n \quad \text{|| production of morphogen}$$

|| Follow morphogen gradient, subject to drag and compression:

$$\mathbb{D}\mathbf{v} = \beta \nabla C - \gamma \cdot \mathbf{v} - n^{-1} \nabla \phi$$

$$\mathbb{D}n = -\nabla \cdot (n\mathbf{v}) + \mathbf{v} \cdot \nabla n \quad \text{|| change of density in material frame}$$

$$\phi = [(n - n_0)^+]^3 \quad \text{|| arbitrary penalty function}$$

## 5 CONCLUSIONS

We have outlined the requirements for a formalism — indeed a programming notation — for embodied computation oriented toward morphogenesis, and have discussed our approach to meeting these requirements. Fundamental to this is massively parallel programming built on a foundation of continuum mechanics and partial differential equations, which facilitates scaling up to very large numbers of embodied computing units. We also presented several examples of morphogenetic programming inspired by embryological morphogenesis. Much additional work remains to be done in order to complete the definition of the formalism, including its rules of inference, and to evaluate its usefulness for programming morphogenesis, which are our current activities.

## REFERENCES

- [1] Andy Adamatzky, Ben De Lacy Costello, and Tetsuya Asai. (2005). *Reaction-Diffusion Computers*. Elsevier, Amsterdam.
- [2] D. Ambrosi, A. Gamba, and G. Serini. (2004). Cell directional persistence and chemotaxis in vascular morphogenesis. *Bulletin of Mathematical Biology*, 67:1851–1873.
- [3] R. D. Barish, P. W. K. Rothmund, and E. Winfree. (2005). Two computational primitives for algorithmic self-assembly: Copying and counting. *Nano Letters*, 5:2586–92.
- [4] R. Benzi, G. Parisi, A. Sutera, and A. Vulpiani. (1982). Stochastic resonance in climatic change. *Tellus*, 34(10–16).
- [5] D. A. Beysens, G. Forgacs, and J. A. Glazier. (2000). Cell sorting is analogous to phase ordering in fluids. *Proc. Nat. Acad. Sci. USA*, 97:9467–71.
- [6] Eric Bonabeau. (1997). From classical models of morphogenesis to agent-based models of pattern formation. *Artificial Life*, 3:191–211.
- [7] Eric Bonabeau, Marco Dorigo, and Guy Theraulaz. (1999). *Swarm Intelligence: From Natural to Artificial Intelligence*. Santa Fe Institute Studies in the Sciences of Complexity. Addison-Wesley, New York.
- [8] Rodney Brooks. (1991). Intelligence without representation. *Artificial Intelligence*, 47: 139–59.
- [9] Scott Camazine, Jean Louis Deneubourg, Nigel R. Franks, James Sneyd, Guy Theraulaz, and Eric Bonabeau. (2001). *Self-Organization in Biological Systems*. Princeton University Press, Princeton.
- [10] Andy Clark. (1997). *Being There: Putting Brain, Body, and World Together Again*. MIT Press, Cambridge, MA.
- [11] Andy Clark and David J. Chalmers. (1998). The extended mind. *Analysis*, 58(7):10–23.
- [12] J. Cooke and E. C. Zeeman. (1976). A clock and wavefront model for control of the number of repeated structures during animal morphogenesis. *J. Theoretical Biology*, 58:455–76.
- [13] P. G. de Gennes. (1992). Soft matter. *Science*, 256:495–497.
- [14] J. L. Deneubourg. (1977). Application de l'ordre par fluctuation à la description de certaines étapes de la construction du nid chez les termites. *Insectes Sociaux*, 24:117–30.
- [15] M.-L. Dequéant and O. Pourquié. (2008). Segmental patterning of the vertebrate embryonic axis. *Nature Reviews Genetics*, 9:370–82.
- [16] Hubert L. Dreyfus. (1979). *What Computers Can't Do: The Limits of Artificial Intelligence*. Harper & Row, New York, revised edition.

- [17] Gabor Forgacs and Stuart A. Newman. (2005). *Biological Physics of the Developing Embryo*. Cambridge University Press, Cambridge, UK.
- [18] Y. C. Fung. (1993). *Biomechanics: Mechanical Properties of Living Tissues*. Springer-Verlag, New York.
- [19] A. Gamba, D. Ambrosi, A. Coniglio, A. de Candia, S. Di Talia, E. Giraudo, G. Serini, L. Preziosi, and F. Bussolino. (March 21 2003). Percolation, morphogenesis, and Burgers dynamics in blood vessels formation. *Physical Review Letters*, 90(11):118101–1 – 118101–4.
- [20] A. Gierer and H. Meinhardt. (1972). A theory of biological pattern formation. *Kybernetik*, 12:30–39.
- [21] Scott F. Gilbert. (2000). *Developmental Biology*. Sinauer Associates, Sunderland, MA, 6th edition.
- [22] C. Gomez, E. M. Özbudak, J. Wunderlich, D. Baumann, J. Lewis, and O. Pourquié. (2008). Control of segment number in vertebrate embryos. *Nature*, 454:335–9.
- [23] Donna Jeanne Haraway. (1976/2004). *Crystals, Fabrics, and Fields: Metaphors that Shape Embryos*. North Atlantic Books, Berkeley, CA.
- [24] S. Haykin. (1999). *Neural Networks: A Comprehensive Foundation*. Prentice Hall, Upper Saddle River, NJ, second edition.
- [25] F. Iida, R. Pfeifer, L. Steels, and Y. Kuniyoshi. (2004). *Embodied Artificial Intelligence*. Springer-Verlag, Berlin.
- [26] M. Johnson and T. Rohrer. (2007). We are live creatures: Embodiment, American pragmatism, and the cognitive organism. In J. Zlatev, T. Ziemke, R. Frank, and R. Dirven, editors, *Body, Language, and Mind*, volume 1, pages 17–54. Mouton de Gruyter, Berlin.
- [27] O. Khatib. (1986). Real-time obstacle avoidance for manipulators and mobile robots. *International Journal of Robotics Research*, 5:90–9.
- [28] S. Kirkpatrick, C.D. Gelatt, Jr., and M.P. Vecchi. (1983). Optimization by simulated annealing. *Science*, 220:671–80.
- [29] Julian Lewis. (2008). From signals to patterns: Space, time, and mathematics in developmental biology. *Science*, 322:399–403.
- [30] Bruce J. MacLennan. (2002). Universally programmable intelligent matter: Summary. In *IEEE Nano 2002 Proceedings*, pages 405–8. IEEE Press.
- [31] Bruce J. MacLennan. (2003). Molecular combinatory computing for nanostructure synthesis and control. In *IEEE Nano 2003 Proceedings*, page 13\_01 ff. IEEE Press.
- [32] Bruce J. MacLennan. (June 2004). Natural computation and non-Turing models of computation. *Theoretical Computer Science*, 317(1–3):115–145.
- [33] Bruce J. MacLennan. (2004). Radical reconfiguration of computers by programmable matter: Progress on universally programmable intelligent matter — UPIM report 9. Technical Report CS-04-531, Dept. of Computer Science, University of Tennessee, Knoxville.
- [34] Bruce J. MacLennan. (August 6 2008). Aspects of embodied computation: Toward a reunification of the physical and the formal. Technical Report UT-CS-08-610, Department of Electrical Engineering and Computer Science, University of Tennessee, Knoxville, Knoxville.
- [35] Bruce J. MacLennan. (2009). Computation and nanotechnology (editorial preface). *International Journal of Nanotechnology and Molecular Computation*, 1(1):i–ix.
- [36] Bruce J. MacLennan. (2009). Super-Turing or non-Turing? Extending the concept of computation. *International Journal of Unconventional Computing*, 5(3–4):369–387.
- [37] Bruce J. MacLennan. (November 15 2010). Molecular coordination of hierarchical self-assembly. Technical Report UT-CS-10-662, Department of Electrical Engineering and Computer Science, University of Tennessee, Knoxville.
- [38] Bruce J. MacLennan. (2010). Morphogenesis as a model for nano communication. *Nano Communication Networks*, 1(3):199–208.

- [39] Bruce J. MacLennan. (in press).  both informed and transformed: Embodied computation and information processes. In Gordana Dodig-Crnkovic and Mark Burgin, editors, *Information and Computation*. World Scientific.
- [40] P. K. Maini and H. G. Othmer, editors. (2001). *Mathematical Models for Biological Pattern Formation*. Springer-Verlag.
- [41] Hans Meinhardt. (1982). *Models of Biological Pattern Formation*. Academic Press, London.
- [42] Richard Menary, editor. (2010). *The Extended Mind*. MIT Press.
- [43] Roeland M.H. Merks, Sergey V. Brodsky, Michael S. Gologorsky, Stuart A. Newman, and James A. Glazier. (2006). Cell elongation is key to in silico replication of in vitro vasculogenesis and subsequent remodeling. *Developmental Biology*, 289:44–54.
- [44] M. I. Miller, B. Roysam, K. R. Smith, and J. A. O’Sullivan. (1991). Representing and computing regular languages on massively parallel networks. *IEEE Transactions on Neural Networks*, 2:56–72.
- [45] G. M. Odell, G. Oster, P. Alberch, and B. Burnside. (1981). The mechanical basis of morphogenesis. I. Epithelial folding and invagination. *Developmental Biology*, 85:446–62.
- [46] R. Pfeifer and J. C. Bongard. (2007). *How the Body Shapes the Way We Think — A New View of Intelligence*. MIT Press, Cambridge, MA.
- [47] R. Pfeifer, M. Lungarella, and F. Iida. (2007). Self-organization, embodiment, and biologically inspired robotics. *Science*, 318:1088–93.
- [48] Rolf Pfeifer and Christian Scheier. (1999). *Understanding Intelligence*. MIT Press, Cambridge, MA.
- [49] E. Rimon and D. E. Koditschek. (1989). The construction of analytic diffeomorphisms for exact robot navigation on star worlds. In *Proceedings of the 1989 IEEE International Conference on Robotics and Automation, Scottsdale AZ*, pages 21–6, New York. IEEE Press.
- [50] P. W. K. Rothmund, N. Papadakis, and E. Winfree. (2004). Algorithmic self-assembly of DNA Sierpinski triangles. *PLoS Biology*, 2(12):2041–53.
- [51] P. W. K. Rothmund and E. Winfree. (2000). The program-size complexity of self-assembled squares. In *Symposium on Theory of Computing (STOC)*, pages 459–68, New York. Association for Computing Machinery.
- [52] I. Salazar-Ciudad, J. Jernvall, and S. A. Newman. (2003). Mechanisms of pattern formation in development and evolution. *Development*, 130:2027–37.
- [53] Guido Serini, Davide Ambrosi, Enrico Giraudo, Andrea Gamba, Luigi Preziosi, and Federico Bussolino. (2003). Modeling the early stages of vascular network assembly. *The EMBO Journal*, 22(8):1771–1779.
- [54] O. Steinbeck, A. Tóth, and K. Showalter. (1995). Navigating complex labyrinths: Optimal paths from chemical waves. *Science*, 267:868–71.
- [55] M. S. Steinberg and T. J. Poole. (1982). Liquid behavior of embryonic tissues. In R. Bellairs and A. S. G. Curtis, editors, *Cell Behavior*, pages 583–607. Cambridge University Press, Cambridge, UK.
- [56] Larry A. Taber. (2004). *Nonlinear Theory of Elasticity: Applications in Biomechanics*. World Scientific, Singapore.
- [57] Guy Theraulaz and Eric Bonabeau. (1995). Coordination in distributed building. *Science*, 269:686–688.
- [58] P.-Y. Ting and R. A. Iltis. (1994). Diffusion network architectures for implementation of Gibbs samplers with applications to assignment problems. *IEEE Transactions on Neural Networks*, 5:622–38.
- [59] Claire J. Tomlin and Jeffrey D. Axelrod. (2007). Biology by numbers: Mathematical modelling in developmental biology. *Nature Reviews Genetics*, 8:331–40.
- [60] A. M. Turing. (1952). The chemical basis of morphogenesis. *Philosophical Transactions of the Royal Society*, B 237:37–72.