Preliminary Development of a Formalism for Embodied Computation and Morphogenesis

Technical Report UT-CS-09-644

Bruce J. MacLennan*

Department of Computer Science
University of Tennessee, Knoxville
www.cs.utk.edu/~mclennan

August 6, 2009

Abstract

The theory of *embodied computation*, like the theory of embodied cognition, provides opportunities as well as challenges. On one hand, such computation is intimately connected with its physical realization, both because post-Moore's Law densities demand more direct exploitation of physical processes, but also because the purposes of embodied computing are often physical (e.g., self-assembly, microrobotics). These characteristics make embodied computing more difficult than conventional computing, because it is not so idealized (independent of its material realization). On the other hand, embodied computation can make productive use of its physical realization, for example, by using the physical states and processes of itself and its environment in place of computational representations. Thus it has implicit computational resources unavailable to conventional computing. In order to fulfill this promise, we will need both formal and informal models of embodied computing that directly address the interaction of formal and physical processes in embodied computational systems. These will be essential cognitive tools for conceptualizing, designing, and reasoning about embodied computation. In this talk I will present a preliminary design for one such model, which is of general applicability, but especially oriented toward *artificial morphogenesis* (self-assembly of complex hierarchical structures by processes analogous to embryological morphogenesis).

1 Background

1.1 Embodied Computation

Pfeifer, Lungarella, and Iida [41, p. 1088] 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. It includes computational processes that directly exploit physical processes for computational ends and those in which information representations and processes 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* [6, 8, 21, 40], but extends them to all computation [30].

We believe that computational processes occurring in nature are well enough understood at this time to begin to abstract some of its central characteristics and to construct a formalism to facilitate the understanding and design of embodied computation systems. The development of such a formalism is the central goal of the proposed research.

1.1.1 Advantages

There are a number of reasons for investigating embodied computing (details can be found elsewhere [30]). 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 ICs). 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 [30, 31, 32]. In part this will be accomplished by developing new physical systems and processes, but the other half of the equation is to develop new models of computation that are closer to the laws of physics.

Another significant advantage of embodied computing is that many computations may be performed "for free" by the physical substrate. For example, as is well known, many artificial neural networks are based on matrix-vector multiplications combined with simple nonlinear functions, such as the logistic sigmoid, $1/(1+e^{-x})$, and many universal approximation theorems are based on linear combinations of sigmoids and

similar functions [20, pp. 208–94]. Whereas computing a sigmoid on a conventional computer requires computing a series approximation to a transcendental function (e.g., exp, tanh) or approximating the sigmoid by table look-up and interpolation, sigmoidal behavior is typical of many physical systems, for it results from an exponential growth process that eventually saturates. For example, available chemical receptors may become occupied or the supply of signaling molecules may become exhausted. In general, sigmoidal response comes for free because physical resources become saturated or depleted. In embodied computing we do not need to program sigmoid functions explicitly; we can exploit common physical processes with the required behavior.

Further, many self-organizing systems depend on positive feedback for growth and extension and on negative feedback for stabilization, delimitation, separation, and the creation of structure (in space or time). In embodied computation negative feedback may be implemented by naturally occurring physical processes such as leakage, evaporation, dispersion, dissipation, and degradation of matter or energy. These processes will occur anyway; embodied computation makes productive use of them.

Some algorithms (e.g., simulated annealing [23], stochastic resonance [3]) use randomness for productive purposes, including escape from local optima, symmetry breaking, deadlock avoidance, exploration, etc. Such randomness comes for free in physical systems in the form of noise, uncertainty, imprecision, and other stochastic phenomena.

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. An 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 [24]. Diffusion can be used for many computational processes, including broadcasting information and massively parallel search, such as in path planning through mazes, optimization, and constraint satisfaction [22, 37, 42, 47, 51]. 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 [7]. 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 [13, 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 presences 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 [13, 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 [7]. The best-known example is wasp nest building [5]. 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.

The greatest degree of integration between a computation and its realization occurs when the computation is not controlling some separate physical system, but is rather modifying or constructing the physical realization of itself. That is, the computer and the computation co-create each other. This is the basis of embryological morphogenesis, in which embodied computation creates the physical substrate for later embodied computation. Cells signal each other in order to coordinate the creation and differentiation of new cells, which extend the morphogenetic process. Further, in later developmental stages, neural processes create the nervous system, including the brain. (Thus living systems are described as *autopoietic*, or *self-making* [34, 38].) Similarly, in some DNA-based algorithmic self-assembly processes, molecular computation creates the physical structure that supports further computation and assembly [2, 43, 44].

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 [26, 27, 28, 29]. A related, but very important,

property is self-repair, since acceptable configurations can be defined as stationary states to which the system reconfigures after damage. Finally, embodied computation systems can be designed for self-destruction, which is especially important for nanoscale systems. If they can *re*configure themselves, they can also *de*configure 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.

In summary, with conventional computing technology we "torture" the physical substrate so that it implements desired computations (e.g., using continuous electronic processes to implement binary logic), whereas embodied computation "respects the medium," conforming to physical characteristics rather than working against them. The goal in embodied computation is to exploit the physics, not to circumvent it (which is costly).

1.1.2 Disadvantages

One of the challenges of embodied computation is our lack of experience with it. Much of our programming has been done in the idealized worlds of perfect logic and implementation-independent programming languages; unavoidable interactions with physical reality have been relegated to the periphery. Fortunately nature provides numerous examples of effective embodied computation, from intracellular genetic regulatory circuitry to the swarm intelligence of social insects and other animals. Therefore we can look to nature to learn how computation can cooperate with physics, rather than opposing it, and how information processing systems can fruitfully interact with the physical embodiment of themselves and other systems.

Traditionally, a sort of Cartesian dualism has reigned in computer science; programs and algorithms have been conceived as idealized mathematical objects; software has been developed, explained, and analyzed independently of hardware; the focus has been on the *formal* rather than the *material*. Embodied computing, in contrast, because of its greater assimilation to physics, is less idealized, less independent of its physical realization. This will increase the difficulty of programming since it will be dependent on (or, some might say, contaminated by) physical concerns. One critical issue is energy. Most embodied computation systems, and especially those incorporating active agents, are dissipative systems, which must be fueled by energy in some form (including chemical reactants) and must dissipate energy and waste material. Contrary to conventional models of computation, a model of embodied computation must address both the acquisition of energy and raw materials, and the dissipation of energy and material wastes. In particular, embodied computation systems must be able to convey energy and raw materials from their sources to the places where they are used, and to convey wastes out of the system without disrupting its activity.

This increased dependence on physical properties might seem to turn computation into a kind of applied physics, but there is still an important role for computational

abstractions. We can see this from the history of contemporary computing technology, for the same mathematical abstraction — Boolean logic — has been used as a model of computation since Boole's *Investigation of the Laws of Thought* (1854), through successive generations of implementation technology, from the mechanical logic of Jevon's logical piano (1869), through relays, vacuum tubes, discrete transistors, integrated circuits, and several generations of VLSI. This stable theoretical background has permitted a cumulative investment in Boolean logic and circuit design, providing continuity from one technological generation to the next, and saving us from having to reinvent computer design with each new technology. This is possible because Boolean logic is physically realizable, yet sufficiently abstract that it can be realized by a variety of physical systems.

Therefore, in laying the foundation for embodied computing we should seek new models of computation that combine physical realism with sufficient abstractness to be implementable in a variety of physical media. Our models of computation need to be close to the underlying physical realization, but not so close that only one realization is possible. Therefore we should adopt as fundamental computational operations those processes that occur in a wide variety of physical systems or that can be fairly directly implemented in them. For example, diffusion is a common physical process, which occurs in a variety of media, from charge carriers diffusing in a semiconductor, to molecules diffusing in a fluid, to cells wandering randomly in a developing embryo, and it has proved useful for information processing and control in natural and artificial systems; therefore it is a good candidate as an operation in embodied computing.

Fortunately nature provides many examples of the use of physical processes for information processing, and these can often be abstracted from their specific physical embodiment and realized in other physical systems. Examples include insect nest building, slime mold aggregation, excitable media and reaction-diffusion systems used to control spatial organization, molecular regulatory circuits in cells, intracellular DNA/RNA computing, and embryological pattern formation and morphogenesis. Understanding these systems in information processing terms will show how common physical processes may be exploited to more directly realize information-processing functions in intimate interaction with their physical realizations, and thus show the way to embodied computing technologies. Indeed, even a model that is ultimately rejected as an account of a biological process might be applied usefully in an artificial embodied computation system.

1.2 Focal Application: Artificial Morphogenesis

Enormous progress has been made in recent years in the nanostructuring of materials, and a variety of techniques are available for fabricating bulk materials with a desired nanostructure. However, the higher levels of organization have been neglected, and nanostructured materials are assembled into macroscopic structures using techniques

that are not essentially different from those used for conventional materials. For example, nanostructured materials may be shaped by machining or molding and assembled by conventional manufacturing techniques. Thus we may have self-assembly at the nanoscale and conventional manufacturing at the macroscale, but no systematic fabrication technology applicable to all scales. Is there an alternative?

Fortunately nature provides a suggestive example, for embryological morphogenesis creates highly complex hierarchical systems, with structures ranging from the nanoscale within cells up through multicellular tissues to the level of gross anatomy. As a significant example, we may take the mammalian nervous system. The brain comprises a number of anatomical regions (the lobes), each comprising hundreds of smaller functional regions (e.g., Brodmann's areas, computational maps), which are structured into macrocolumns, which in turn contain minicolumns, each with a half-dozen or so layers. The minicolumns comprise about one hundred neurons with dendritic trees of characteristic shape (and tens of thousands of synapses), all interconnected in specific ways. At the other end of the scale, the brain itself is part of a nervous system, which includes a highly ramified but organized network of nerves. Thus, embryological morphogenesis provides an inspiring example of how self-organized growth, differentiation, and interaction can produce these complex macroscopic structures from microscopic components. Similarly, the mathematical principles of morphogenesis may be applicable to the fabrication of complex hierarchically-structured artificial systems. The physical realization of these mathematical principles is closely connected to computation.

Morphogenesis has a number of distinct 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 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 [49, ch. 6]. In other cases, tissues behave more like viscous fluids, perhaps percolating through a more solid matrix, and

this fluid motion is essential in cell migration [4][13, pp. 92–4][48]. Non-cellular substances, such as morphogens and other signaling chemicals diffuse like gases through non-isotropic media, but cells also exhibit facilitated diffusion [13, pp. 13–15, 156, 252]. In many cases, tissues occupy a middle ground, with viscoelastic properties [13, pp. 21–2, 133]. In general terms, morphogenesis takes place in the relatively unexplored realm of *soft matter* [10][13, 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* [50]. 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 [9, 12, 18]), and morphogenesis may be best understood as the creation of patterns in *four* dimensions [17, p. 504n].

Developmental biologists have identified a number of fundamental processes involved in morphogenesis [13, pp. 158–9][45] (for an enumeration, see Sec. 4, p. 25). 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

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

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. 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). On the other hand, while some of these phenomena are *physical continua* (e.g., electrical fields), others are *phenomenological continua* composed of microscopic discrete elements (atoms, molecules, cells, microrobots, etc.).

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 elements, we will want to be able to move fluently between the two perspectives. This is especially the case in morphogenesis, where in the early stages of development we are faced with discrete phenomena — 1, 2, 4, 8, etc. cells with specific shapes and arrangements — whereas 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 [13].

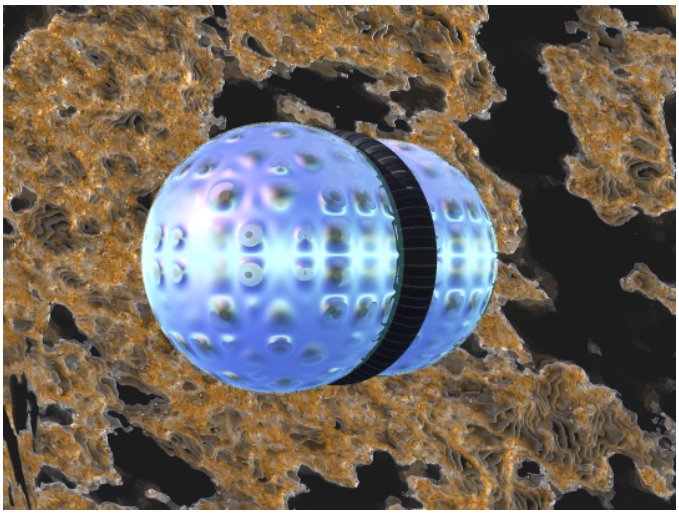

Figure 1: Artistic depiction of microrobot.

Therefore we propose that a formalism for embodied computation (especially as applied to morphogenesis) 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 elements, 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 finite, discrete units in a finite, discrete set. 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.

To maintain these complementary interpretations, the formalism should treat the elements as having a small, indeterminate size. To maintain this indeterminacy, 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. For example, shape should be treated in an intensive way. Thus we should not think of the elements as differential volume elements with a parallelepiped shape ($\mathrm{d}x\,\mathrm{d}y\,\mathrm{d}z$), which may not be appropriate for discrete particles (e.g., cells). Nevertheless, the 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

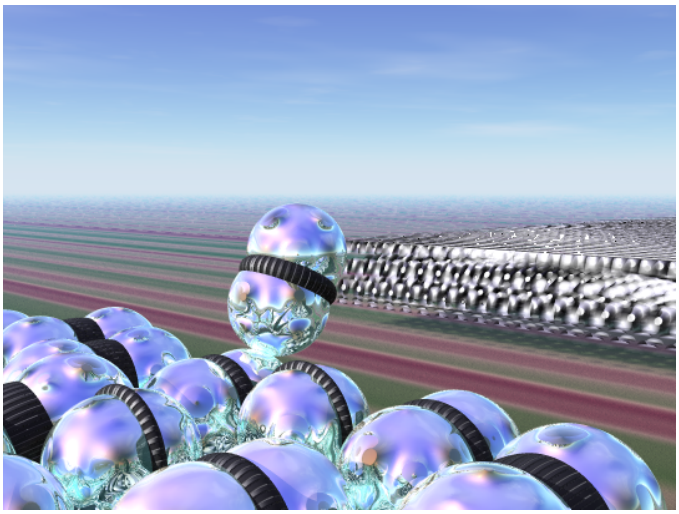

Figure 2: Artistic depiction of microrobots self-assembling to form artificial tissues.

in all kinds of media, including solids, liquids, gases, and physical fields. Further, we know that viscoelastic media (so-called "soft matter" [10]) are important in morphogenesis (for example, mesenchyme is best characterized as viscoelastic [13, p. 98][14]), so 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 [19]. 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 naturally 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 it 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 and curvilinear coordinates seem to be the most suitable terms in which 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.

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.

Biologists often find it convenient to express regulatory networks qualitatively. For example, they may use *influence models* to indicate that one gene product enhances or represses the expression of another gene [52]. These qualitative regulatory relationships are an important tool for conceptualizing control processes in which the quantitative relationships are unknown or are not considered critical (e.g., the same final state results from a wide range of parameter values). Therefore, the formalism should permit the expression of qualitative control relationships, and the simulator should permit their execution (e.g., by supplying default functions and parameters and facilitating their variation).

Of course a central goal of the formalism is that it support complementary discrete and continuous descriptions of substances, tissues, processes, etc., because this complementarity is valid in the realm of many important applications of embodied computation, where large numbers of microscopic elements constitute macroscopic bodies. Nevertheless, discrete and continuous models sometimes lead to different results (e.g., for small cell numbers). Therefore the simulator should permit simulations to be run with various meshes and discretizations to ensure that results are not artifacts of the simulation parameters.

A related issue is that multiple space and time scales will be relevant to complex embodied computation systems, such as those implementing morphogenesis of complex hierarchical structures. Thus a phenomenon that appears discrete at one scale might be better treated as continuous at a larger scale. Conversely, objects and events that seem discrete at a large scale, may appear continuous at a finer scale. Since a complete embodied computation system may involve subsystems at many scales, the formalism must permit their integration into a single description.

In embodied computation, especially as it pushes towards the nanoscale, noise, error, uncertainty, defects, faults, and other sources of indeterminacy and unpredictability are unavoidable [25]. 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, 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 Approach

Having outlined the requirements for a formalism for embodied computation, we present a preliminary definition of such a formalism. The goal of the proposed research is the further development and evaluation of this formalism.

3.1 Description

3.1.1 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.

Multiple inheritance, allowing a class to be subclass of two or more disjoint classes and to inherit the attributes of both of them, is an idea that has proved useful in object-oriented programming and is supported by some object-oriented programming languages. Nevertheless it has some complications, including consistency problems. Multiple inheritance is potentially useful in embodied computing, and we intend to explore it as well as some alternatives.

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 in Sec. 3.1.2, 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 (tensor) 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. (Examples of substance definitions are given below, Sec. 3.2.)

3.1.2 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*).[1] Typically bodies are homogeneous, that is, composed of elements of a par-

[1] In continuum mechanics, "particle" and "material point" are synonymous, but we use "particle" when it is most natural to think of the body as composed of discrete units, and "material point" when it is natural to think of it as a physical continuum. The distinction is entirely heuristic.

ticular 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, is 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 *identitifiers* for the elements. At any given time t each element $P \in \mathcal{B}$ has a location in a region $\mathcal{R}_t \subset \mathcal{E}$ of three-dimensional Euclidean space[2] defined by a vector $\mathbf{p} = C_t(P)$, where $C_t : \mathcal{B} \to \mathcal{E}$ is a continuous function that maps $\mathcal{B}$ into $\mathcal{E}$ with range $\mathcal{R}_t = C_t[\mathcal{B}]$. 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.

$\mathcal{B}$ has a topology homeomorphic to $\mathcal{R}_t$, and in fact it is often convenient to label the particles by their position at some distinguished time t_0, such as the initial preparation of the system. In this case, C_{t_0} is an identity function. When the particles are labeled by their position in some such reference configuration, we write $\mathbf{P} \in \mathcal{B} = \mathcal{R}_{t_0}$ to emphasize that the particle labels are vectors in $\mathcal{E}$.

The configuration mapping C_t is generally taken to be a diffeomorphism; in particular, it is continuous, invertible, and sufficiently differentiable for the purpose at hand. This common assumption would seem to present a problem, since a body must remain homeomorphic to its previous states, which presents bodies from dividing, joining, or changing topological genus, and would seem to limit morphogenesis. There are at least two solutions to this problem. One is to relax the requirement that C_t be a diffeomorphism, which is not always required. The second, and better, answer is to realize that bodies are abstract unities, effectively manifolds in which (as will be explained in Sec. 3.1.3) various properties, such as number density, can vary. As a consequence, the number density of elements in them can arbitrarily small. If the density becomes zero between two regions of a body, the body will be effectively divided, and similarly, if a zero density becomes positive, then separate regions may be joined. This is in fact exactly what we need in morphogenesis, in which changes in number density of cell types define tissues. For examples, consider the reaction-diffusion systems and the model of vasculogenesis discussed below (Secs. 3.2.2, 3.2.4),

[2]We limit our formulation to three-dimensional space because our focal application is morphogenesis, the creation of three-dimensional form. Occasionally we consider the simpler cases of one- and two-dimensional pattern formation.

both of which are capable of generating high-genus structures.

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 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 undivided cell), the definition of a uniform gravitational or electrical field, 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.

3.1.3 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 the 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, $\mathbf{p}$: $\mathbf{V}(P) = \mathrm{D}_t\mathbf{p} = \mathrm{D}_t C_t(P)$. 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). Even when properties are commonly described in spatial terms (e.g., a background temperature or electrical field), we prefer, when possible, to use a material description by attaching the properties to a (perhaps rigid and immovable) substrate. Nevertheless, the best ways of describing embodied computations will emerge from experience using

the formalism.

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* $\mathrm{Kn} = \lambda/L$ is the ratio of a particle's mean free path length λ 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 genetic regulatory circuits 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, $\mathbf{U}$.

More generally, many embodied computation processes make use of vectors fields (e.g., the gradients of scalar fields) to guide the motion of particles. Examples include chemical concentration gradients and electrical gradients. Mathematically, each point of the field has a well-defined vector value, but to maintain complementarity we must associate with each element a vector-valued random variable (or a probability density function over vectors).

Embodied computation systems often depend on the movement of masses of particles, and this is especially the case in morphogenesis, and so we are often concerned with the *flux* of some intensive quantity. For example, the mass-density flux $\rho\mathbf{v}$ repre-

sents the direction and rate at which mass density is moving. Similarly, the number-density flux $n\mathbf{v}$ represents the movement of particle density. The local decrease in density is then given by the divergence of the flux, $\nabla \cdot (n\mathbf{v})$.

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 [13, pp. 113–16][39]. 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. 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 reliably manipulate 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.

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.

Although the requirements for this formalism dictate that embodied computation be expressed in terms of intensive quantities, there may be circumstances in which extensive quantities are unavoidable. One solution that we are exploring is the use of several fundamental *free extensive variables*. For example, δV may represent the (possibly infinitesimal) volume of an element of some body, so that the (extensive quantity) m, the mass of the element, is given by $m = \rho \delta V$, where ρ is the (intensive) mass density. The variable δV is called a *free* extensive variable because it has no

specific value, but it obeys certain formal rules of manipulation (analogous to differential notation in ordinary calculus). Other free extensive variables might include the area and diameter of elements (δA, δL respectively), perhaps with free scale factors to account for indefinite element shapes. Naturally, these need to be manipulated carefully to maintain mathematical consistency, but it is our intention that they will be needed only rarely. In any case, this is an experimental feature, which we will explore.

3.1.4 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., $\mathrm{D}_t X = F(X, Y)$, 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 Δt is implicit in the Δ_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: $\mathrm{D}_t X = \partial X/\partial t|_{P\,\mathrm{fixed}}$. 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:

$$\text{Đ}X = F(X, Y).$$

This *change equation* may be read, "the change in X is given by $F(X, Y)$." (The *edh operator* Đ is provisional notation.) The rules of manipulation for the Đ operator respect its complementary 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.

As discussed under Sec. 2, noise, uncertainty, error, faults, defects, and other sources of randomness are a given in embodied computation, especially when applied in nanotechnology. Indeed, such randomness can often be exploited as a computational resource 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:

$$\text{Đ}X = F(X, Y, \ldots) + G_1(X, Y, \ldots)\nu_1(t) + \cdots + G_n(X, Y, \ldots)\nu_n(t),$$

where the ν_j are noise terms,[3] but we must be careful about the interpretation of equations of this kind.

To see why, consider a stochastic integral, $X_t = \int_0^t H_s \mathrm{d}W_s$, where W is a Wiener process (Brownian motion). In differential form this is $\mathrm{d}X_t = H_t \mathrm{d}W_t$. 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 Itō calculus. The corresponding stochastic change equation in Langevin form is $Ð X_t = H_t Ð W_t$, but we must be careful about its interpretation. 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, $\mathrm{D}_t X_t = H_t \mathrm{D}_t W_t$, is problematic, since a Wiener process is nowhere differentiable. Fortunately we can treat $\mathrm{D}_t W$ purely formally, as follows. First observe that $\Delta W_t = W_{t+\Delta t} - W_t$ is normally distributed with zero mean and variance Δt, $\Delta W_t \sim \mathcal{N}(0, \Delta t)$. Therefore $\Delta_t W_t = \Delta W_t / \Delta t$ is normally distributed with unit variance, $\Delta_t W_t \sim \mathcal{N}(0, 1)$, and $\Delta_t W_t$ can be treated as a random variable with this distribution. To extend this to the continuous case, we treat $Ð W_t$ formally as a random variable, $Ð W_t \sim \mathcal{N}(0, 1)$. Therefore the stochastic change equation $Ð X = H Ð W$ has consistent complementary interpretations. Similarly, for an n-dimensional Wiener process $\mathbf{W}^n$, we interpret $Ð \mathbf{W}^n$ 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[4]

$$Ð X \sim -X, Y, Z$$

(for example) to mean that the change of X is "repressed" (negatively regulated) by X and "enhanced" (positively regulated) by Y and Z. We have been calling such a relationship a *regulation* (as opposed to an *equation*) and read it, "the change in X is negatively regulated by X and positively regulated by Y and Z." Formally, $Ð X \sim -X, Y, Z$ is interpreted as a change equation $Ð X = F(-X, Y, Z)$ in which F is a function that is unspecified except that it is monotonically non-decreasing in each of its arguments. (Thus the signs of the arguments express positive or negative regulation.) Therefore, the general form of a regulation is

$$Ð X_k \sim \pm X_1, \pm X_2, \ldots, \pm X_n.$$

Scaling might be applied to the arguments to express relative strengths of regulation (e.g., $Ð X \sim -\eta X, Y, Z$, where $0 < \eta \leq 1$), but that is already beginning to deviate from the spirit of a qualitative relationship. Another notation that we are exploring is to separate parameters by the approximate order of magnitude of their influence

[3] Frequently the G_j are constant functions, in which case the noise is *additive*.

[4] This use of the "$\sim$" symbol is unrelated to its use to indicate the distribution of a random variable.

strength, for example, "$\text{Đ}X \sim Y, Z; -X$" means that the influence strength of $-X$ is qualitatively less than that of Y and Z. In an influence diagram, the parameters separated by semicolons might be represented by arrows of different widths. Our research will explore the use of these notations in various applications to determine which are most useful.

3.2 Examples

We consider, very briefly, several simple examples of the formalism.

3.2.1 Diffusion

Diffusion is an Itō process in an m-dimensional diffusion medium:

$$\text{Đ}\mathbf{p} = \boldsymbol{\mu} + \boldsymbol{\sigma}\,\text{Đ}\mathbf{W}^n,$$

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 $\mathbf{W}^n$ 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 [13, 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):

$$\text{Đ}\phi = \nabla \cdot [-\boldsymbol{\mu}\phi + \nabla \cdot (\boldsymbol{\sigma}\boldsymbol{\sigma}^{\mathrm{T}}\,\phi)/2].$$

Hence, $\boldsymbol{\mu}$ represents the average particle velocity, and the rank-2 tensor $\boldsymbol{\sigma}\boldsymbol{\sigma}^{\mathrm{T}}/2 \in \mathbb{R}^{m\times m}$ 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:

scalar field ϕ	‖ concentration
vector fields:	
$\mathbf{j}$	‖ flux
$\boldsymbol{\mu}$	‖ drift vector
order-2 **field** $\boldsymbol{\sigma}$	‖ diffusion tensor

behavior:

$$\begin{aligned} \mathbf{j} &= \boldsymbol{\mu}\phi - \nabla \cdot (\boldsymbol{\sigma}\boldsymbol{\sigma}^{\mathrm{T}}\,\phi)/2 \quad \| \text{ flux} \\ \text{Đ}\phi &= -\nabla \cdot \mathbf{j} \quad \| \text{ change in concentration} \end{aligned}$$

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** $-1 \leq Z \leq 1$:

$$\begin{aligned} \mathbf{j} &= 0 \quad \| \text{ initial 0 flux} \\ \boldsymbol{\mu} &= 0 \quad \| \text{ drift vector} \\ \boldsymbol{\sigma} &= 0.1\mathbf{I} \quad \| \text{ isotropic diffusion} \end{aligned}$$

for $X^2 + Y^2 + Z^2 \leq 0.1$:

$$\phi = 100 \quad \| \text{ initial density inside sphere}$$

for $X^2 + Y^2 + Z^2 > 0.1$:

$$\phi = 0 \quad \| \text{ zero density outside sphere}$$

3.2.2 Activator-Inhibitor System

Activator-inhibitor systems are two-component reaction-diffusion systems that have proved useful as models of pattern formation in development [16, 33, 35, 53]. (See Fig. 3.) Both components diffuse, the activator A promoting the production of both, the inhibitor I repressing production:

$$\begin{aligned} \text{Đ}A &\sim A, -I, \triangle(\boldsymbol{\sigma}_A\boldsymbol{\sigma}_A^{\mathrm{T}}\,A), \\ \text{Đ}I &\sim A, -I, \triangle(\boldsymbol{\sigma}_I\boldsymbol{\sigma}_I^{\mathrm{T}}\,I), \end{aligned}$$

where $\triangle = \nabla \cdot \nabla$ is the Laplacian operator on tensor fields. A more complete description, in the notation of the formalism, 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

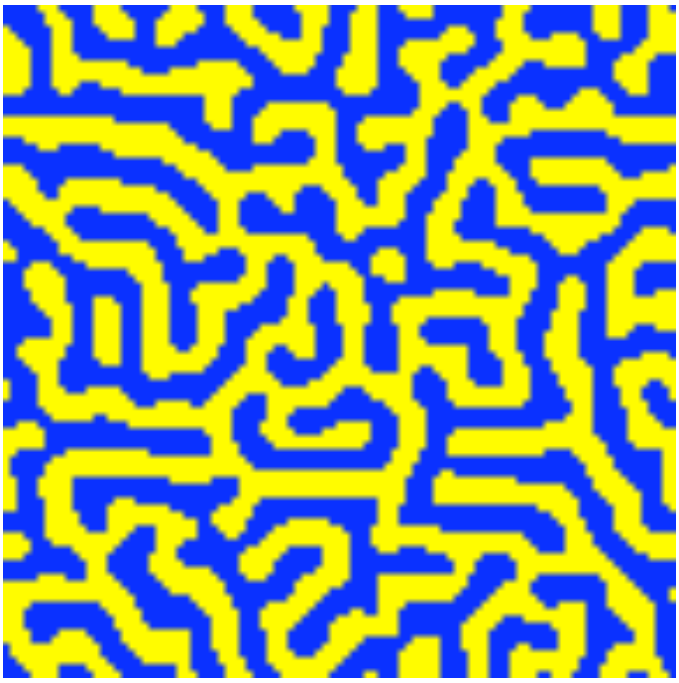

Figure 3: Typical pattern formed by activator-inhibitor system.

behavior:

$$\begin{aligned} \text{Đ}A &\sim A, -I, \triangle(\boldsymbol{\sigma}_A \boldsymbol{\sigma}_A^{\mathrm{T}} A) \\ \text{Đ}I &\sim A, -I, \triangle(\boldsymbol{\sigma}_I \boldsymbol{\sigma}_I^{\mathrm{T}} I) \end{aligned}$$

To use specific reaction functions f_A and f_I (defined elsewhere) we write:

behavior:

$$\begin{aligned} \text{Đ}A &= f_A(A, I) + \triangle(\boldsymbol{\sigma}_A \boldsymbol{\sigma}_A^{\mathrm{T}} A) \\ \text{Đ}I &= f_I(A, I) + \triangle(\boldsymbol{\sigma}_I \boldsymbol{\sigma}_I^{\mathrm{T}} I) \end{aligned}$$

3.2.3 Deneubourg Model

Deneubourg's model of pillar construction in wasp nests provides a good example of a continuous model of a discrete process [11]. 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 concentration
scalars:

k_1	‖	deposition rate
k_2	‖	pheromone evaporation rate

behavior:

$$\text{Đ}C \;=\; k_1 T - k_2 C \quad \| \text{ deposition - evaporation}$$

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 ϕ degrades at rate k_3 and diffuses subject to $\mathbf{D}_\phi$:

substance pheromone **is** cement **with**:

scalar field ϕ	‖	concentration in air
scalar k_3	‖	degradation rate
order-2 **field** $\mathbf{D}_\phi$	‖	diffusion tensor field

behavior:

$$\text{Đ}\phi \;=\; k_2 C - k_3 \phi + \triangle(\mathbf{D}_\phi \phi) \quad \| \text{ evaporation from cement - degradation + diffusion}$$

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

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

scalar field T	‖	density
vector field $\mathbf{v}$	‖	velocity field
scalar r_T	‖	flux of cement-laden termites into system (a constant)
order-2 **field** $\mathbf{D}_T$	‖	diffusion (wandering) tensor field
scalar k_4	‖	strength of gradient following

behavior:

$$\begin{aligned}
\mathbf{v} &= k_4 \nabla\phi - T^{-1}\nabla \cdot \mathbf{D}_T T && \| \text{ gradient following - diffusion} \\
\text{Đ}\mathbf{p} &= \mathbf{v} && \| \text{ change in position} \\
\text{Đ}T &= r_T - k_1 T - \nabla \cdot (T\mathbf{v}) + \mathbf{v} \cdot \nabla T && \| \text{ change in concentration in material frame}
\end{aligned}$$

Recall that $\mathbf{p}$ (spatial position) is a property of all substances, and so the equation for Đ$\mathbf{p}$ 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 ϕ, which is a property of the substance phereomone. 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.

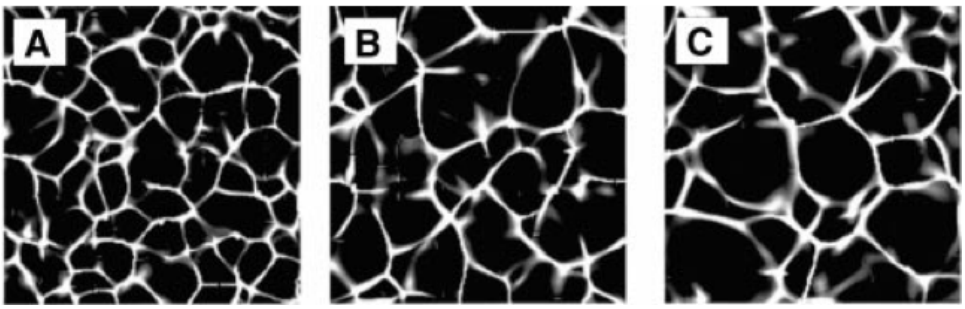

Figure 4: Examples of simulated vasculogenesis for various cell densities [46].

3.2.4 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 [1, 15, 46] (cf. [36]). (See Fig. 4.) Aggregation is governed by a morphogen that is released by the cells and that diffuses and degrades.

substance morphogen **is** medium **with**:

scalar fields:

C	‖ concentration
S	‖ source
order-2 **field D**	‖ diffusion tensor
scalar τ	‖ degradation time constant

behavior:

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

The cells produce morphogen at a rate α and follow the gradient at a rate given by β. Cell motion is impeded by dissipative interactions with the substrate, which is measured by an order-2 tensor field γ. 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 a function, ϕ, that increases very rapidly as the density exceeds n_0.

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

scalar fields:

n	‖ number density of cell mass
ϕ	‖ cell compression force

vector field v ‖ cell velocity
scalars:
n_0 ‖ maximum cell density
α ‖ rate of morphogenesis release
β ‖ strength of morphogen attraction
order-2 field γ ‖ dissipative interaction

behavior:

$$
\begin{aligned}
S &= \alpha n \quad \| \text{ production of morphogen} \\
&\quad \| \text{ follow morphogen gradient, subject to drag and compression:} \\
\text{Đ}\mathbf{v} &= \beta \nabla C - \gamma \cdot \mathbf{v} - n^{-1} \nabla \phi \\
&\quad \| \text{ change of density in material frame:} \\
\text{Đ}n &= -\nabla \cdot (n \cdot \mathbf{v}) + \mathbf{v} \cdot \nabla n \\
\phi &= [(n - n_0)^+]^3 \quad \| \text{ arbitrary penalty function}
\end{aligned}
$$

4 Pattern Forming Processes

A principal goal of the project is to demonstrate that the formalism is sufficiently expressive to describe the fundamental processes of embryological development [13, pp. 158–9][45] and to demonstrate them with the simulator. We will be using models and descriptions from the embryological development literature. These developmental processes fall into three classes:

(I) In the strict sense, *morphogenesis* refers to the creation of three dimensional forms with modification of cell state. These processes include *directed mitosis* (division of cells with a consistent orientation), *differential growth* (leading to deformation of tissues), *apoptosis* (programmed cell death altering forms), *differential adhesion* (leading to cell sorting and compartment formation), *condensation* (e.g., of cells embedded in mesenchyme), *contraction* (with consequent stress-induced deformation), *matrix modification* (through swelling, degradation, and other physical processes), and *migration* of various kinds, including *diffusion* (undirected movement), *chemokinesis* (migration speed subject to an ambient chemical cue), *chemotaxis* (migration governed by a gradient in chemical morphogen or substrate), and *haptotaxis* (motion through differential adhesion to a substrate).

(II) Another important mechanism of pattern formation is *modification of cell states*, which leads to differentiation of cell behaviors and properties. On one hand, *cell autonomous mechanisms* do not depend on interactions with other cells; they depend on both spatial nonuniformities (i.e., *asymmetric mitosis*, in which the two daughter cells have different properties) and temporal nonuniformities (i.e., *mitosis with temporal dynamics*, in which oscillation asynchronized with the cell cycle leads to

development of spatial patterns). On the other hand, *inductive mechanisms* depend on cell-cell signalling, which may be *hierarchic* (involving unidirectional signalling) or *emergent* (involving feedback through mutual induction).

(III) Finally, there are *morphodynamic mechanisms*, which combine induction and morphogenesis, but are poorly understood [45].

5 Conclusions

We have outlined the requirements for a formalism for embodied computation oriented toward morphogenesis, and have discussed our approach to meeting these requirements. Much additional work remains to be done in order to formally define the formalism, including its rules of inference, which is our current activity.

References

[1] D. Ambrosi, A. Gamba, and G. Serini. Cell directional persistence and chemotaxis in vascular morphogenesis. *Bulletin of Mathematical Biology*, 67:1851–1873, 2004.

[2] R. D. Barish, P. W. K. Rothemund, and E. Winfree. Two computational primitives for algorithmic self-assembly: Copying and counting. *Nano Letters*, 5:2586–92, 2005.

[3] R. Benzi, G. Parisi, A. Sutera, and A. Vulpiani. Stochastic resonance in climatic change. *Tellus*, 34(10–16), 1982.

[4] D. A. Beysens, G. Forgacs, and J. A. Glazier. Cell sorting is analogous to phase ordering in fluids. *Proc. Nat. Acad. Sci. USA*, 97:9467–71, 2000.

[5] Eric Bonabeau, Marco Dorigo, and Guy Theraulaz. *Swarm Intelligence: From Natural to Artificial Intelligence*. Santa Fe Institute Studies in the Sciences of Complexity. Addison-Wesley, New York, 1999.

[6] Rodney Brooks. Intelligence without representation. *Artificial Intelligence*, 47:139–59, 1991.

[7] Scott Camazine, Jean Louis Deneubourg, Nigel R. Franks, James Sneyd, Guy Theraulaz, and Eric Bonabeau. *Self-Organization in Biological Systems*. Princeton University Press, Princeton, 2001.

[8] Andy Clark. *Being There: Putting Brain, Body, and World Together Again*. MIT Press, Cambridge, MA, 1997.

[9] J. Cooke and E. C. Zeeman. A clock and wavefront model for control of the number of repeated structures during animal morphogenesis. *J. Theoretical Biology*, 58:455–76, 1976.

[10] P. G. de Gennes. Soft matter. *Science*, 256:495–497, 1992.

[11] J. L. Deneubourg. 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, 1977.

[12] M.-L. Dequéant and O. Pourquié. Segmental patterning of the vertebrate embryonic axis. *Nature Reviews Genetics*, 9:370–82, 2008.

[13] Gabor Forgacs and Stuart A. Newman. *Biological Physics of the Developing Embryo*. Cambridge University Press, Cambridge, UK, 2005.

[14] Y. C. Fung. *Biomechanics: Mechanical Properties of Living Tissues*. Springer-Verlag, New York, 1993.

[15] A. Gamba, D. Ambrosi, A. Coniglio, A. de Candia, S. Di Talia, E. Giraudo, G. Serini, L. Preziosi, and F. Bussolino. Percolation, morphogenesis, and Burgers dynamics in blood vessels formation. *Physical Review Letters*, 90(11):118101–1 – 118101–4, March 21 2003.

[16] A. Gierer and H. Meinhardt. A theory of biological pattern formation. *Kybernetik*, 12:30–39, 1972.

[17] Scott F. Gilbert. *Developmental Biology*. Sinauer Associates, Sunderland, MA, 6th edition, 2000.

[18] C. Gomez, E. M. Özbudak, J. Wunderlich, D. Baumann, J. Lewis, and O. Pourquié. Control of segment number in vertebrate embryos. *Nature*, 454:335–9, 2008.

[19] Donna Jeanne Haraway. *Crystals, Fabrics, and Fields: Metaphors that Shape Embryos*. North Atlantic Books, Berkeley, CA, 1976/2004.

[20] S. Haykin. *Neural Networks: A Comprehensive Foundation*. Prentice Hall, Upper Saddle River, NJ, second edition, 1999.

[21] M. Johnson and T. Rohrer. 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, 2007.

[22] O. Khatib. Real-time obstacle avoidance for manipulators and mobile robots. *International Journal of Robotics Research*, 5:90–9, 1986.

[23] S. Kirkpatrick, C.D. Gelatt, Jr., and M.P. Vecchi. Optimization by simulated annealing. *Science*, 220:671–80, 1983.

[24] Julian Lewis. From signals to patterns: Space, time, and mathematics in developmental biology. *Science*, 322:399–403, 2008.

[25] Bruce J. MacLennan. Natural computation and non-Turing models of computation. *Theoretical Computer Science*, 317(1–3):115–145, June 2004.

[26] Bruce J. MacLennan. Radical reconfiguration by universally programmable intelligent matter. In *Air Force Reconfigurable Electronics Workshop*, Santa Fe, NM, May 17–18 2004. Air Force Research Laboratory.

[27] Bruce J. MacLennan. 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, 2004.

[28] Bruce J. MacLennan. Highly programmable matter and generalized computation: Research in reconfigurable analog and digital computation in bulk materials. In *1st AFRL Reconfigurable Systems Workshop*, Albuquerque, NM, Feb. 14–15 2007. Air Force Research Laboratory.

[29] Bruce J. MacLennan. Self-organization for nano-computation and nano-assembly. In *Workshop on Emergent Behavior (WEB 07)*, Oak Ridge, TN,, March 6–7 2007. Oak Ridge National Laboratory.

[30] Bruce J. MacLennan. Embodiment and non-Turing computation. In *2008 North American Computing and Philosophy Conference: The Limits of Computation*, Bloomington, IN, July 10–12 2008. The International Association for Computing and Philosophy. http://www.cs.utk.edu/~mclennan/papers/AEC-TR.pdf.

[31] Bruce J. MacLennan. Computation and nanotechnology (editorial preface). *International Journal of Nanotechnology and Molecular Computation*, 1(1):i–ix, 2009.

[32] Bruce J. MacLennan. Super-Turing or non-Turing? Extending the concept of computation. *International Journal of Unconventional Computating*, 5(3–4):369–387, 2009.

[33] P. K. Maini and H. G. Othmer, editors. *Mathematical Models for Biological Pattern Formation*. Springer-Verlag, 2001.

[34] H. Maturana and F. Varela. *Autopoiesis and Cognition: The Realization of the Living*, volume 42 of *Boston Studies in the Philosophy of Science*. D. Reidel Publishing Co., Dordecht, 1980. R. S. Cohen and M. W. Wartofsky (Eds.).

[35] Hans Meinhardt. *Models of Biological Pattern Formation*. Academic Press, London, 1982.

[36] Roeland M.H. Merks, Sergey V. Brodsky, Michael S. Goligorksy, Stuart A. Newman, and James A. Glazier. Cell elongation is key to in silico replication of in vitro vasculogenesis and subsequent remodeling. *Developmental Biology*, 289:44–54, 2006.

[37] M. I. Miller, B Roysam, K. R. Smith, and J. A. O'Sullivan. Representing and computing regular languages on massively parallel networks. *IEEE Transactions on Neural Networks*, 2:56–72, 1991.

[38] J. Mingers. *Self-Producing Systems*. Springer, New York, 1994.

[39] G. M. Odell, G. Oster, P. Alberch, and B. Burnside. The mechanical basis of morphogenesis. I. Epithial folding and invagination. *Developmental Biology*, 85:446–62, 1981.

[40] R. Pfeifer and J. C. Bongard. *How the Body Shapes the Way We Think — A New View of Intelligence*. MIT Press, Cambridge, MA, 2007.

[41] R. Pfeifer, M. Lungarella, and F. Iida. Self-organization, embodiment, and biologically inspired robotics. *Science*, 318:1088–93, 2007.

[42] E. Rimon and D. E. Koditschek. 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, 1989. IEEE Press.

[43] P. W. K. Rothemund, N. Papadakis, and E. Winfree. Algorithmic self-assembly of DNA Sierpinski triangles. *PLoS Biology*, 2(12):2041–53, 2004.

[44] P. W. K. Rothemund and E. Winfree. The program-size complexity of self-assembled squares. In *Symposium on Theory of Computing (STOC)*, pages 459–68, New York, 2000. Association for Computing Machinery.

[45] I. Salazar-Ciudad, J. Jernvall, and S. A. Newman. Mechanisms of pattern formation in development and evolution. *Development*, 130:2027–37, 2003.

[46] Guido Serini, Davide Ambrosi, Enrico Giraudo, Andrea Gamba, Luigi Preziosi, and Federico Bussolino. Modeling the early stages of vascular network assembly. *The EMBO Journal*, 22(8):1771–1779, 2003.

[47] O. Steinbeck, A. Tóth, and K. Showalter. Navigating complex labyrinths: Optimal paths from chemical waves. *Science*, 267:868–71, 1995.

[48] M. S. Steinberg and T. J. Poole. Liquid behavior of embryonic tissues. In R. Bellairs and A. S. G. Curtis, editors, *Cell Behavior*, pages 583–607. Cambridge University Press, Cambridge, UK, 1982.

[49] Larry A. Taber. *Nonlinear Theory of Elasticity: Applications in Biomechanics*. World Scientific, Singapore, 2004.

[50] Guy Theraulaz and Eric Bonabeau. Coordination in distributed building. *Science*, 269:686–688, 1995.

[51] P.-Y. Ting and R. A. Iltis. Diffusion network architectures for implementation of Gibbs samplers with applications to assignment problems. *IEEE Transactions on Neural Networks*, 5:622–38, 1994.

[52] Claire J. Tomlin and Jeffrey D. Axelrod. Biology by numbers: Mathematical modelling in developmental biology. *Nature Reviews Genetics*, 8:331–40, 2007.

[53] A. M. Turing. The chemical basis of morphogenesis. *Philosophical Transactions of the Royal Society*, B 237:37–72, 1952.