Chapter 11
Embryomorphic Engineering: Emergent Innovation Through Evolutionary Development

René Doursat, Carlos Sánchez, Razvan Dordea, David Fourquet and Taras Kowaliw

Abstract *Embryomorphic Engineering*, a particular instance of Morphogenetic Engineering, takes its inspiration directly from biological development to create new robotic, software or network architectures by decentralized self-assembly of elementary agents. At its core, it combines three key principles of multicellular embryogenesis: chemical gradient diffusion (providing positional information to the agents), gene regulatory networks (triggering their differentiation into types, thus patterning), and cell division or aggregation (creating structural constraints, thus reshaping). This chapter illustrates the potential of Embryomorphic Engineering in different spaces: 2D/3D physical swarms, which can find applications in collective robotics, synthetic biology or nanotechnology; and *n*D graph topologies, which can find applications in distributed software and peer-to-peer techno-social networks. In all cases, the specific genotype shared by all the agents makes the phenotype's complex architecture and function modular, programmable and reproducible.

This chapter is a condensed review version of references [16–22].

R. Doursat (✉) · T. Kowaliw
Complex Systems Institute Paris Ile-de-France (ISC-PIF),
CNRS and Ecole Polytechnique, 57-59, rue Lhomond, 75005 Paris, France
e-mail: rene.doursat@polytechnique.edu

C. A. Sánchez
Research Group in Biomimetics (GEB),
Universidad de Málaga, C/ Severo Ochoa 4, PTA, 29590 Malaga, Spain

R. Dordea · D. Fourquet
Erasmus Mundus Masters in Complex Systems Science (EMMCS),
Ecole Polytechnique, Route de Saclay, 91120 Palaiseau, France

R. Doursat et al. (eds.), *Morphogenetic Engineering*, Understanding Complex Systems,
DOI: 10.1007/978-3-642-33902-8_11,

11.1 Evolutionary Development

Morphogenetic Engineering (ME), the topic of this book, concerns the design, or rather "meta-design", of the self-organizing abilities of the elements of complex systems toward functional architectures. This meta-design, however, should not exclusively rely on human inventiveness as in traditional engineering disciplines but may also involve an important automation part, essentially via an evolutionary search. In that sense, by combining not only self-organization and architecture but also evolution, ME is very close to the tenets of *evolutionary development*, a recent and rapidly expanding field of biology nicknamed "evo-devo" [6, 8, 10, 11, 30, 40, 48, 58, 68].

11.1.1 Evo-Devo in Biology

In the variation/selection couple of evolutionary biology, "selection" has received most of the honors while "variation" remained the neglected child. Darwin discovered the evolution of species, based on random mutations and nonrandom natural selection, and established it as a central fact of biology. During the same period, Mendel brought to light the laws of inheritance of traits. In the twentieth century, his work was rediscovered and became the foundation of the science of genetics, which culminated with the revelation of DNA's role in heredity by Avery and its double-helix structure by Watson and Crick. Integrating evolution and genetics, the "Modern Synthesis" of biology has successfully demonstrated the existence of a fundamental correlation between genotype and phenotype and between their respective changes: mutation in the first is causally related to variation in the second. Yet, 150 years after Darwin's and Mendel's era, the nature of the link from genes to organismal forms, i.e., the actual molecular and cellular basis of the *mechanisms* of development, are still unclear. *How* does a one-dimensional genome specify a three-dimensional animal? [24]. *How* does a static, linear DNA unfold in time (regulation dynamics) and space (cellular self-assembly)? What is the part also played by epigenetics? These questions constitute the missing link of the Modern Synthesis and the main challenge of evo-devo.

While the attention was focused on selection, it is only during the past decade that analyzing and understanding variation (as the generation of phenotypic innovation) by comparing the *developmental* processes of different species, at both the embryonic and the genomic levels, became again a major concern of biology. Researchers realized that the genotype-phenotype pairing could not forever remain an abstraction if they wanted to understand the unique power of evolution to produce countless innovative structures—and, concerning Artificial Life and bio-inspired engineering, ultimately transfer this understanding to self-organized technological systems. To quote Kirschner and Gerhart [40, p. ix]:

> When Charles Darwin proposed his theory of evolution by variation and selection, explaining selection was his great achievement. He could not explain variation. That was Darwin's

> dilemma. . . . To understand novelty in evolution, we need to understand organisms down to their individual building blocks, down to their deepest components, for these are what undergo change.

Evo-devo casts a new light on the question still little addressed by today's predominant gene-centric view of biology: To what extent are organisms also the product of complex physicochemical developmental processes not necessarily or always controlled by complex underlying genetics? Before and during the advent of genetics, the study of developmental structures had been pioneered by the "structuralist" school of theoretical biology, which can be traced back to Goethe, D'Arcy Thompson, and Waddington. Later, it was most actively pursued and defended by Kauffman [38, 39] and Goodwin [30] under the banner of *self-organization*, argued to be an even greater force than natural selection in the production of viable diversity.

Recent dramatic advances in the genetics and evolution of biological development have paved the way toward explaining morphological self-organization and sketching an encompassing "generativist" theory of embryogenesis. The objective is to unify organisms beyond their seemingly "endless forms most beautiful" (in the words of Darwin [7]) by unraveling the generic mechanisms that make them variations around a common theme [68]. The variations are the specifics of the genetic and epigenetic information; the theme is the *developmental dynamics* that this information steers. It comprises the elementary laws by which the genome produces the very proteins that can further interpret it, controlling cell division, differentiation, adhesion and death, and ultimately producing an anatomy. On this keyboard, evolution is the ultimate player.

11.1.2 Evo-Devo in Artificial Life

Looking at the full evolutionary *and* developmental picture should also be a primary concern of systems engineering and computer science when venturing into the new arena of autonomous, distributed architectures. Evolutionary Computation (EC) techniques such as genetic algorithms or genetic programming, which were inspired by evolutionary biology in its traditional modern-synthesis form, have like their natural model principally focused on selection through virtual "genomic operators", "fitness functions" and "reproduction rates". As a consequence, the great majority of these approaches rely on more or less direct and abstract mappings from artificial genomes to artificial individuals, while including only little or no morphogenesis.

Therefore, one important goal of a new field of "Alife evo-devo" is to provide the computational foundation for a virtual re-engineering of the "strongly morphogenetic" complex systems spontaneously produced by nature, such as biological development. To this aim, one must design a programmable and reproducible two-way indirect mapping between the local rules of self-assembly followed by the elementary agents at the microscopic level (the genotype Γ), and the collective structure and function of the system at the macroscopic level (the phenotype Φ). Calculating

the transformation from Γ to Φ corresponds to developing an organism—while solving the inverse problem of finding an appropriate Γ given a desired Φ (or family of similar Φ's), would be the challenge of an evolutionary search, whether goal-oriented, open-ended, or a mix of the two.

Mirroring the evo-devo paradigm in biological systems, new EC avenues need to stress the importance of fundamental laws of developmental variations as a prerequisite to selection on the evolutionary time scale of artificial systems [62]. From the EC viewpoint, it means an implicit or *indirect* mapping from genotype to phenotype. Fine-grained, hyperdistributed architectures similar to multicellular organisms (i.e., many light-weight agents, as opposed to a few heavy-weight agents) might be in a unique position to provide the "solution-rich" space needed for successful selection and *spontaneous innovation through developmental modularity and composition*.

11.1.3 From Embryogenesis to Embryomorphic Engineering

Putting in practice the above theoretical intentions, this chapter offers an overview of a recent framework called *Embryomorphic Engineering*, founded in 2006 by René Doursat [16, 17] (who coined the term after "Neuromorphic Engineering") to explore the causal and programmable link from genotype to phenotype that is needed in many emerging computational disciplines, such as artificial embryogeny [5, 46, 62], and apply it to innovative uses. Its endeavors as a bio-inspired computing technology follow those of biological evo-devo, and for this reason it could be equivalently referred to as "Evo-Devo Engineering". Embryomorphic Engineering works on two levels in parallel: it consists of simultaneous genetic engineering (Γ) and functional shape engineering (Φ), based on a common playground made of a multitude of small agents capable of self-assembling into a particular organism. These agents are guided by the genetic instructions they carry, which parametrize and modulate the fundamental laws of biomechanical-like assembly and biochemical-like signaling that they obey, creating appropriate context-sensitive rules.

The remainder of the text illustrates the potential of Embryomorphic Engineering in different spaces: 2D/3D physical swarms, which can find applications in collective robotics, synthetic biology or nanotechnology; and *n*D graph topologies, which can find applications in distributed software and peer-to-peer techno-social networks. In all cases, the specific genotype shared by the agents makes the phenotype's complex architecture and function modular, programmable and reproducible:

- Section 11.2 describes *MapDevo* (Modular Architecture by Programmable Development), the original and foundational 2D model of embryonic development based on self-assembly, pattern formation, and genetic regulation (Fig. 11.1). Section 11.3 examines hand-made mutations of the genotypes of MapDevo organisms and their corresponding phenotypes, paving the way toward an evolutionary version of programmable development. It is followed in Sect. 11.4 by a study of *functional*—not merely morphological—architectures, called *fMapDevo*, through

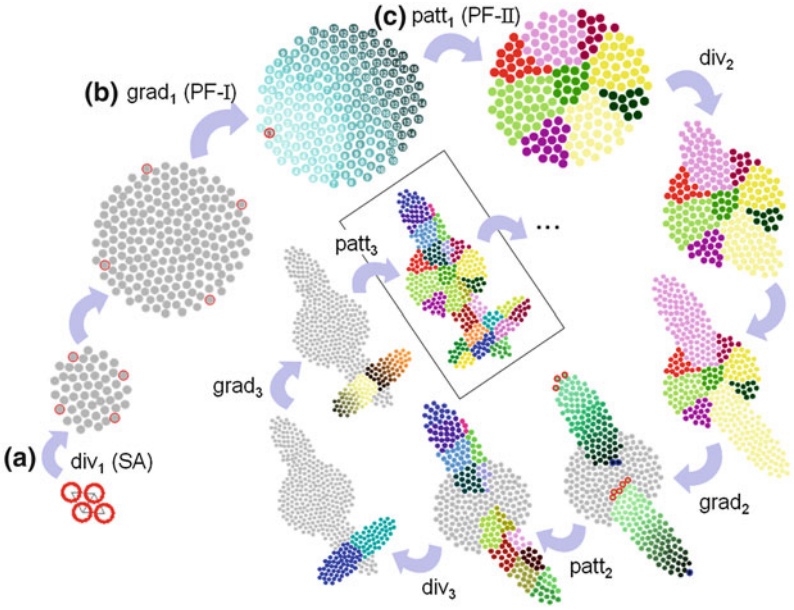


Fig. 11.1 The iterative 3-stage MapDevo growth cycle

a model of animated embryomorphic organisms immersed in a 3D physical environment.

- After those 2D/3D models, which remained close to their biological inspiration based on multicellular development, Sect. 11.5 presents *ProgNet* (Programmable Network Growth), an extension of MapDevo to *n*D graph topologies via a model of autonomous network construction. There, nodes execute the same program in parallel, communicate and differentiate, while links are dynamically created and removed based on "ports" and "gradients" that guide nodes to specific attachment locations. As the network grows, nodes switch different rules on and off, creating chains, lattices, and other composite topologies. Finally, Sect. 11.6 introduces *ProgLim* (Program-Limited Aggregation), a particular implementation of ProgNet in cellular automata, and Sect. 11.7 briefly concludes the chapter.

11.2 MapDevo: Modular Architecture by Programmable Development

The spontaneous making of an entire organism from a single cell is the epitome of a self-organizing *and* programmable complex system. Through a precise spatiotemporal interplay of genetic switches and chemical gradients, an elaborate form is created without explicit architectural plan or engineering intervention. Embryomorphic

agent-based modeling and simulation attempt to understand and exploit these fundamental morphogenetic mechanisms.

On the one hand, research in *self-assembling* (SA) systems, whether natural or artificial, has traditionally focused on pre-existing components endowed with fixed shapes [69]. Biological development, by contrast, dynamically creates new cells that acquire selective adhesion properties and forms through differentiation induced by their neighborhood [72]. On the other hand, biological *pattern formation* (PF) phenomena [28, 42, 45, 50, 63, 73] are generally construed as orderly states of activity on top of a quasi-continuous and fixed 2D or 3D background of cellular substrate. Yet again, the spontaneous patterning of an organism into regions of gene expression arises within a multicellular medium in perpetual expansion and reshaping. Finally, both phenomena (SA and PF) are often thought of in terms of *stochastic events*—whether mixed components that randomly collide during SA, or spots and stripes that crop up unpredictably from instabilities during PF. Here too, these notions need significant revision if they are to be extended and applied to embryogenesis. Cells are not randomly mixed but pre-positioned where cell division occurs. Genetic identity regions are not randomly distributed but highly *regulated* in number and position.

This section describes *MapDevo* (Modular Architecture by Programmable Development), the original and foundational 2D model of Embryomorphic Engineering first published in [16–18]. It is a spatial computational simulation of *programmable* and *reproducible* morphogenesis that combines SA and PF under the control of a nonrandom gene regulatory network (GRN) stored inside each cell of a swarm. The differential properties of cells (division, adhesion, migration) are determined by the regions of gene expression to which they belong, while at the same time these regions further expand and segment into subregions due to the self-assembly of differentiating cells. To follow an artistic metaphor [10], SA is similar to "self-sculpting" and PF to "self-painting". The model can be construed from two different vantage points: either pattern formation on *moving* cellular automata, in which cells divide and spatially rearrange under the influence of their own activity pattern; or collective motion in a *heterogeneous* swarm, in which cells gradually differentiate and modify their interactions according to their positions and the regions they form.

In the next subsections, the motion of a homogeneous swarm of cells (pure SA) and the patterning by gradient propagation on a static swarm (pure PF) are introduced separately. Then, these two components are combined to form reproducible growing patterns (SA + PF). The genetic control inside every cell guiding these arrangements is also explained. Finally, this combination is repeated in modules ($SA^k + PF^k$) inside a larger, heterogeneous system to create complex morphologies by recursive refinement of details.

Self-assembly by Division and Adhesion (SA) The original MapDevo model consists of a 2D swarm of cells with dynamically changing neighbor interactions calculated by a Delaunay-Voronoi tessellation (Fig. 11.2). Each cell follows two major laws of cellular biomechanics in a simplified format: (i) *cell division*, coded by a uniform probability p for any cell to split into two, and (ii) *cell adhesion*, represented by elastic forces derived from a quadratic potential V with resting length r_e,

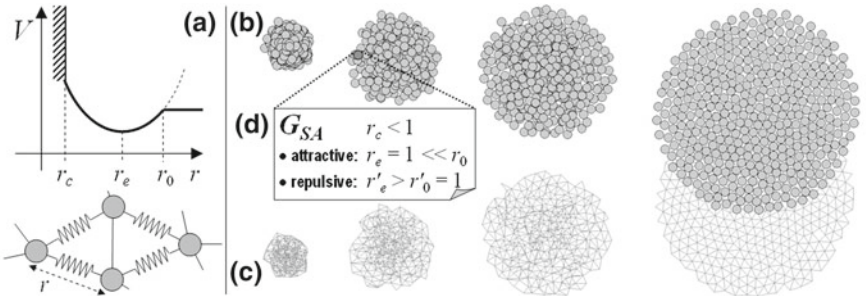


Fig. 11.2 Deployment of a homogeneous swarm (SA). **a** Cell-to-cell interaction potential V similar to elastic springs. **b** Relaxation of a 400-cell swarm from an initially compressed layout. **c** Same swarm viewed from its underlying mesh of pairwise interactions, obtained by Delaunay triangulation and pruning of links longer than r_0. **d** Genetic SA parameters inside every cell (from [18])

hard-core radius r_c, and scope of visibility r_0, similar to collective motion models [31, 66] but with zero velocity (no self-propulsion). These parameters are grouped into a genotype G_{SA}. Laws of motion are derived from a spring-damper system with negligible mass/inertia effects. Under potential V, starting from a compressed swarm, cells quickly relax to a resting state that forms a quasi-regular hexagonal mesh.

Propagation of Positional Information by Gradients (PF-I) Pieces of a jigsaw puzzle are defined not only by their position and shape but also by the "image" that they carry. In our self-organized swarm, this translates into state variables that determine the PF activity inside each cell. The model distinguishes between two kinds of PF-specific state variables (i.e., signals that cells continuously exchange and process): *gradient* variables (PF-I) and *pattern* variables proper (PF-II).

Gradient values (PF-I) propagate from cell to cell to establish *positional information* across the swarm [71]. For example, each cell contains a counter variable g_W. The source cell of this gradient, denoted W, is characterized by $g_W = 0$. It passes value 1 to neighboring cells, which in turn tell their neighbors to set g_W to 2, and so on (Fig. 11.3). To give this isotropic propagation a specific direction, another local rule instructs each cell to retain only the smallest of the current counter value and the received values. The result is a roughly circular wave pattern of increasing g_W counters centered on source W. Together with W, three other gradients, E, N and S, contribute to form a 2D coordinate system via equatorial (midline) axes $X = NS^{\perp}$ and $Y = WE^{\perp}$, which contain the cells where counter values cross, respectively: $|g_Y = g_N - g_S| \leq 1$ and $|g_X = g_W - g_E| \leq 1$. Note that the four sources W, E, N, S position themselves, too, by "hopping" away from each other (i.e., passing a flag representing source N to any neighbor with a higher S-gradient value, and vice-versa; same for W and E). First defined by Lewis Wolpert [71], "positional information" is a fundamental concept of biological development, and its natural chemical versions (morphogen diffusion and messenger-based signaling) are often translated into discrete counters in artificial systems like this one, such as in Amorphous Computing [12, 49] and Spatial Computing [3, 4].

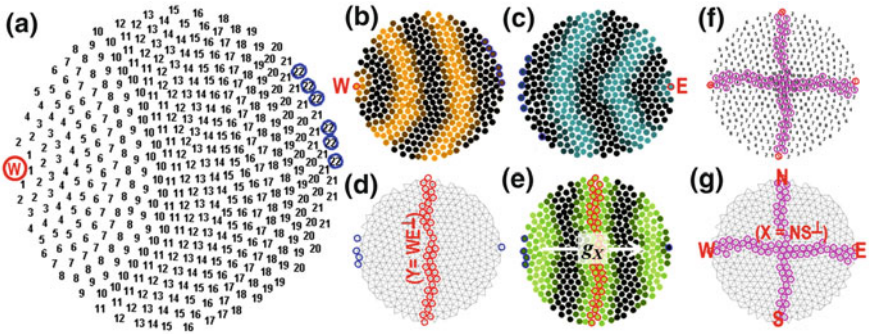


Fig. 11.3 Propagation of positional information (PF-I). **a** Circular gradient of counter values originating from source cell W (circled in *red*; end points circled in *blue*). **b** Same gradient values viewed through a cyclic color map. **c** Opposite gradient coming from antipode cell E. **d** Set of midline cells $Y = WE^{\perp}$ whose W and E counters are equal ± 1. **e** Quasi-planar gradient $g_X = g_W - g_E$. **f**, **g** Full coordinate compass with axis $X = NS^{\perp} \approx WE$ (adapted from [18])

Programmed Patterning by Gene Expression Levels (PF-II) Pattern values (PF-II) correspond to gene expression levels that are calculated on top of the (g_X, g_Y) gradient values to create different *cell types* (which in turn affect the SA behavior; see SA + PF integration below). This calculation relies on a *gene regulatory network* (GRN) inside each cell, whose weights constitute the genetic parameters of the PF process and are denoted by G_{PF} (Fig. 11.4). The inputs of the GRN are the morphogen/second-messenger proteins whose concentrations are encoded in the gradient values g_X and g_Y. Thus the core architecture of the virtual organism is a network of networks, i.e., an irregular 2D lattice of identical GRNs locally coupled to each other via "chemical signaling" nodes (here, g_X and g_Y) [13, 47, 53].

The patterning process represents the emergence of *heterogeneity*, i.e., the segmentation of the swarm into "identity regions" corresponding to high expression levels of particular genes I_k of the GRN. A well-known example is the early striping of *Drosophila* [8] controlled by a 5-layer hierarchy of segmentation genes along the anteroposterior axis (maternal genes, gap genes, primary/secondary pair-rule genes, and segment polarity genes). The present model relies on a 3-layer caricature of the same principle along the two intersecting axes X and Y: (1) the bottom (input) layer of the GRN groups the two positional variables (morphogen concentrations) g_X and g_Y; (2) the middle layer groups "boundary" genes B_i, which segment the embryo into roughly horizontal and vertical half-planes of strong and weak expression levels via 2D step functions σ; (3) the top (output) layer groups the identity genes I_k derived from positive and negative products of the B_i's, i.e., various intersections of the B_i half-planes.

Simultaneous Growth and Patterning (SA + PF) After describing the self-assembly of a non-patterned swarm (SA) and the patterning of a fixed swarm (PF), the embryomorphic SA and PF behaviors are combined to create growing patterns at

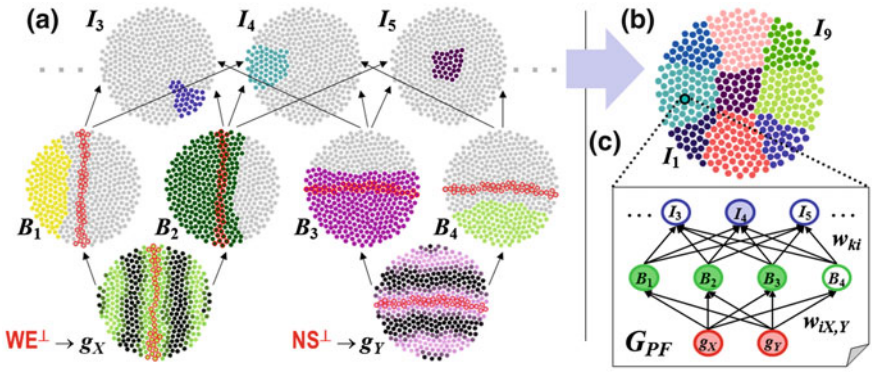


Fig. 11.4 Programmed patterning (PF-II). **a** Same swarm viewed under different color maps revealing the regions where cells' internal variables g_X, g_Y, B_i and I_k are highest (virtual equivalent of *in situ* hybridization in biology). **b** Consolidated view of all identity regions I_k for $k = 1...9$. **c** The GRN, denoted G_{PF}, used by each cell to calculate its expression levels, here: $B_1 = \sigma(-1/3 - g_X)$, $B_3 = \sigma(1/3 - g_Y)$, $I_4 = B_1 B_3(1 - B_4)$, etc. (adapted from [18])

every stage (Fig. 11.5). Cells continually adjust their positions according to the elastic SA constraints, *while* exchanging PF signals over the same dynamic links. This dual dynamics is guided by the combined genotype $G = (G_{\text{SA}}, G_{\text{PF}})$. Daughter cells inherit all the attributes of mother cells, including G and the current internal PF variables (gradient counters and gene levels). As for the SA variables (coordinates and adhesion/signaling links of the lattice), they are recalculated from a position close to the original cell. Both sets of variables are updated as the newborn cell immediately starts contributing to the SA forces and the traffic of PF gradients, which maintain the pattern's consistency at all times in the swarm.

Modular, Recursive Patterning (PF^k) Natural embryological patterns, however, do not develop in one shot but in numerous incremental stages [10]. An adult organism is produced through modular, recursive growth and patterning. In *Drosophila*, regions of the embryo that acquire leg, wing or antenna identity (called "imaginal discs") start developing *local* coordinate systems of morphogen gradients to support the prepatterning and construction of the planned organ [8]. Correspondingly, the present embryomorphic model includes a pyramidal *hierarchy* of network modules able to generate patterns in a recursive fashion (Fig. 11.6). First, the base network G^0_{PF} establishes the main identity regions as above, then subnetworks G^k_{PF} triggered by the identity genes I_k of G^0_{PF} further partition these regions into smaller, specialized compartments at a finer scale. This "trigger" is based on usual gene regulation mechanisms, whereby proteins produced by I_k bind to the regulatory sites of genes B^k_i in the upper layer and are necessary to promote their expression. Fractal patterning has also been explored in generative algorithms such as "L-systems" [52, 61]. These algorithms, however, are most often self-similar and rely on *symbolic* rules and explicit geometry. In contrast, MapDevo is a *dynamical* system of physicochemical

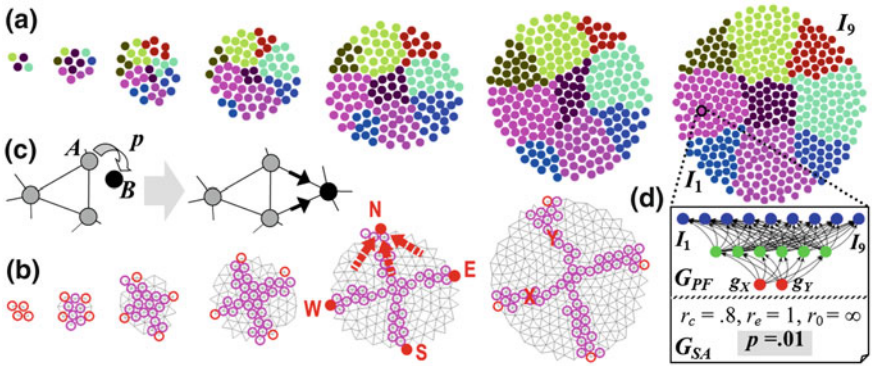


Fig. 11.5 Simultaneous growth and patterning (SA + PF). **a** Swarm growing from 4 to 400 cells by division. **b** Swarm mesh, highlighting gradient sources and midline axes. The gradients and the whole pattern are continually maintained by source migration, e.g., N moves away from S and toward $WE^{\perp}$ (same with the other three sources). **c** Cell B created by A's division quickly contributes to SA forces and PF traffic. **d** Combined genomes inside each cell (adapted from [18])

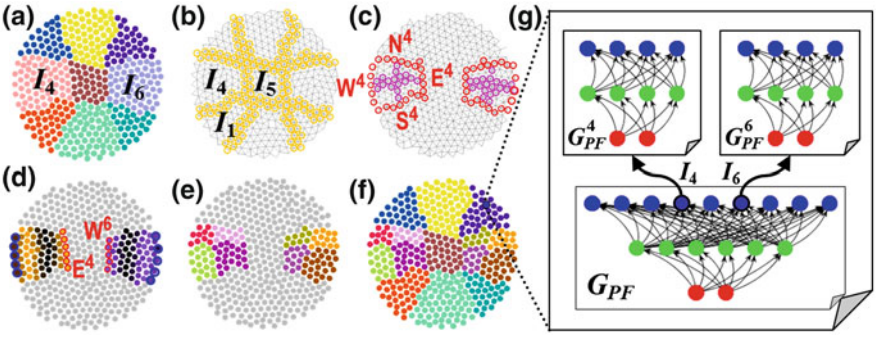


Fig. 11.6 Modular, recursive patterning (PF^k). **a** A 9-region swarm, as in Fig. 11.4b. **b** Cells at the border between two domains are highlighted with *yellow circles*. **c** These border cells become new gradient sources (*red circles*) inside certain identity regions at a lower scale. **d** Missing border sources arise from the ends (*blue circles*) of other gradients. **e**, **f** Subpatterning of the swarm inside I_4 and I_6. **g** Corresponding hierarchical GRN: $G_{\mathrm{PF}} = \{G^0_{\mathrm{PF}}, G^4_{\mathrm{PF}}, G^6_{\mathrm{PF}}\}$ (from [18])

interactions among a multitude of units (a distinction also made in cognitive science between Artificial Intelligence and Neural Networks).

Modular, Anisotropic Growth (SA^k) So far missing from the model is a true topological *deformation* dynamics, or "morphodynamics", that can confer non-trivial shapes to the organic system beyond mere blobs. To this aim, cells must be able to *diversify their SA characteristics, depending on their PF type and spatial position*—thus closing the feedback loop between genetics and geometry [11]. In particular, they have to exhibit *nonuniform, anisotropic* cell division

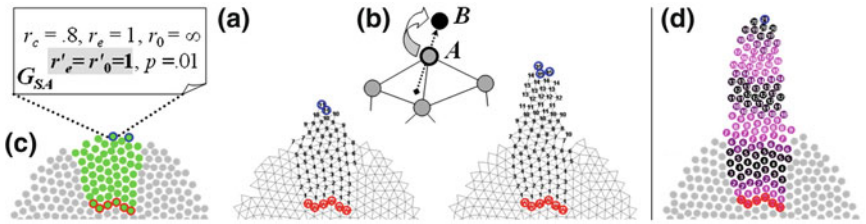


Fig. 11.7 Modular, anisotropic growth (SA^k). **a** Genetic SA parameters are augmented with repelling V values r'_e and r'_0 used between the growing region (*green*) and the rest of the swarm (*gray*). **b** Daughter cells are positioned away from the neighbors' center of mass. **c** Offshoot growth proceeds from an "apical meristem" made of gradient ends (*blue circles*). **d** Cyclic coloring of the gradient underlying this growth (from [18])

(varying p) and *differential adhesion* (varying V). For example, in our artificial model, the growth of limb-like structures can be achieved by a coarse imitation of meristematic plant offshoots (Fig. 11.7). In this process, only the tip or "apical meristem" of the organ is actively dividing at any time (whereby cells forming the tip self-identify as being the local maxima of the gradient generated by the base of the limb). Moreover, potential V is defined to be attractive only among cells within the limb region, while it becomes repelling (i.e., $r_0 \leq r_e$, see Fig. 11.2a) between the limb and other areas. Just like inhomogeneous division, differential adhesion is an essential ingredient of complex shape formation [34, 44].

Modular Growth and Patterning ($SA^k + PF^k$) Putting everything together, full morphologies can develop and self-organize from a few cells (Fig. 11.8). These morphologies are *complex*, *programmable* and *reproducible*: they are architecturally complex because they can be made of any variety of modules and parts that are not necessarily repeated in any periodic or self-similar way; they represent programmable phenotypes because they emerge from a same given genotype carried by every cell of the swarm; they are reproducible, because their structure and shape are not left to chance but tightly controlled by the genotype.

Naturally, the exact positions of the cells at the microscopic level are still random, but not the positions of the mesoscopic and macroscopic regions that they form. Moreover, the modularity of the phenotype is a direct reflection of the modularity of the genotype. The hierarchical SA and PF dynamics recursively unfolds inside the different regions and subregions that it creates. Each module $G^k = (G^k_{\mathrm{SA}}, G^k_{\mathrm{PF}})$ can be reused by exact duplication, but can also diverge from other blocks through different internal genetic SA and PF parameters, potentially giving each region a different morphodynamic behavior and a different gene activity landscape. *Duplication* of gene modules followed by *divergence* of these copies is the basis of *serial homology*, a major evolutionary mechanism in nature exemplified by vertebrae, teeth, or digits [8]. Here, the integration between SA and PF is controlled by the identity

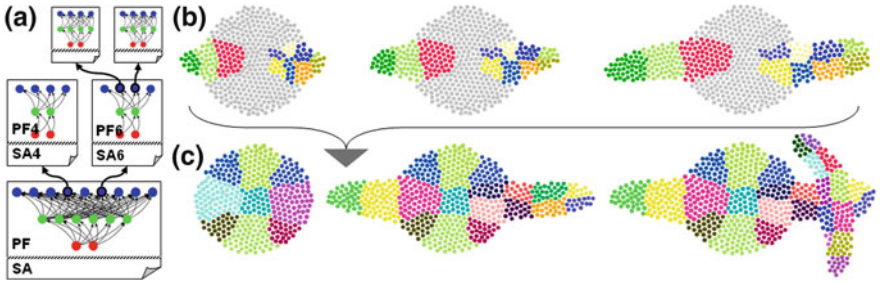


Fig. 11.8 Modular growth and patterning ($SA^k + PF^k$). **a** Example of a three-tier modular genotype giving rise to the artificial organism on the *right*. **b** Three iterations detailing the simultaneous limb-like growth process (Fig. 11.7) *and* patterning of these limbs during execution of the middle tier (modules 4 and 6). **c** Main stages of the complex morphogenesis process, showing full patterns after execution of the bottom, middle and top tiers (from [18])

genes I_k: their corresponding nodes in the GRN switch on the execution of subordinate modules G^k (Fig. 11.6g), i.e., their gene expression activity (parametrized by G^k_{PF}) to create new local segmentation patterns, and their mechanical behavior (parametrized by G^k_{SA}) to create new morphodynamical processes.

11.3 Toward an Evolutionary MapDevo Through Variation

This section presents experiments involving hand-made mutations of the genotypes of MapDevo systems and their corresponding phenotypes (first published in [19]). For now, these systems are purely developmental and do not serve a specific purpose. There is no organism fitness or selection-based evolutionary search. These important aspects are included in ongoing projects, which will be previewed in Sects. 11.4 and 11.6. Here, we exclusively focus on *variation* to illustrate the link between genotype and phenotype, and the programmable and predictable effect that changes in the former can have on the latter via self-organization—in which *modularity* is an essential condition of future evolvability [6, 58, 67].

The figures of this section show several simulation examples of modular embryogenesis and how certain mutations in the genotype correlate with quantitative or qualitative changes in the phenotype. The organism of Fig. 11.9a is taken as the reference or "wild type". Its genotype is composed of two layers with one module each: a base module establishing the *body* plan (lower module) and a specialized module in charge of growing a simple *limb*-like appendage (upper module). The latter is executed twice, in the left and right regions of the body, switched on by identity genes I_4 and I_6 (like in Fig. 11.6a). As just described in Sect. 11.2, each module consists of two types of genomes: a self-assembly genome G_{SA}, encoding how cells divide and spread spatially, and a pattern formation genome G_{PF}, encoding how cells acquire their types. To simplify the figures, the GRN that constitutes G_{PF} is not displayed

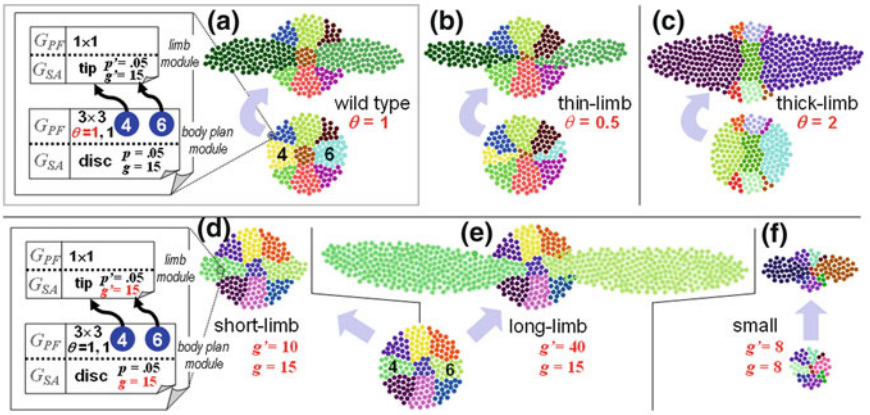


Fig. 11.9 Simulation trials showing quantitative variations. **a**–**c** Varying limb thickness by modifying the GRN weights and thresholds. **d**–**f** Varying length and size of limbs and body by stopping cell division earlier or later (from [19])

in its entirety; instead, only the type of checkered pattern that it produces (explained below) and the module-switching identity genes are sketched.

Quantitative Variation of Limb Thickness by GRN Weights (PF) In Fig. 11.9b, the wild-type organism has been affected by a "thin-limb" mutation of the body plan. Although not shown, some weights and thresholds of the base G_{PF} have been modified in such a way that they now create a checkered pattern with a narrower central row (displacing the B_i gene domains of Fig. 11.4a). This gives less space for the limb buds to grow, hence making them thinner. The reverse, "thick-limb" mutation is shown in Fig. 11.9c, with coefficient 2. This is a good example of the compactness of the developmental genotype [26, 62] and its large-scale effect on the phenotype: slightly varying the sensitivity of a couple of genes can already result in significant morphological changes.

Quantitative Variation of Limb Length by Division Signals (SA) By modifying the division rate and/or the stop conditions of proliferation, the size of various parts of the embryo can also be modulated. For example, in Fig. 11.9d and e, cell proliferation is regulated in the limbs. Here, it is achieved by stopping division when the gradient values of the tip cells (blue circles in Fig. 11.7) reach a specific value g', respectively sooner ($g' = 10$) and later ($g' = 40$) than the wild type ($g = 15$). In Fig. 11.9f, both body plan and limbs stop growing beyond gradient value $g' = 8$, producing a phenotypic shape that is proportionally smaller than the wild type. Note that similar effects can also be achieved by decreasing or increasing the probability of division p, while keeping the stop gradient values constant (see Fig. 11.10c).

Structural Change of Limb Position by Module Switching In Fig. 11.10, the modularity of the limb component is demonstrated through various mutations reminiscent of experiments on biological organisms such as *Drosophila*. The identity genes marking the regions ("imaginal discs") responsible for the growth of a specific

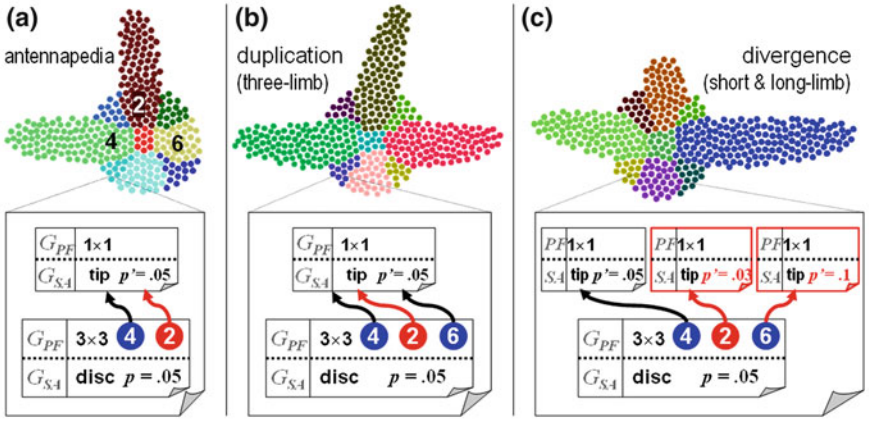
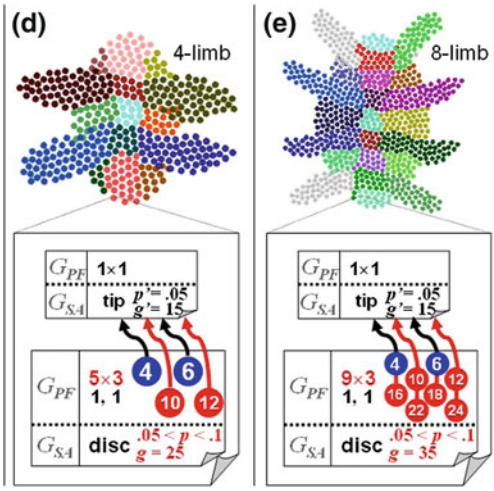


Fig. 11.10 Simulation trials showing structural variations. **a**–**c** Changing limb configuration by switching the limb-triggering genes and/or duplicating the limb module (from [19])

Fig. 11.11 More structural variations. **d**, **e** Adding limbs by body plan expansion (from [19])



appendage [8, 10] can be literally turned on or off in new regions with respect to the wild type of Fig. 11.9a. For example, in Fig. 11.10a, a virtual case of "antennapedia", i.e., the ectopic growth of a leg where there should be an antenna, is obtained by connecting a new identity region to the limb module, here region I_2 instead of region I_6. This means *rewiring* the GRN of G_{PF} to reflect the fact that the regulatory sites of the limb genes on the DNA have mutated and now accept gene I_2's proteins as promoters instead of gene I_6's proteins. In the three-limb mutation of Fig. 11.10b, these regulatory sites have been duplicated before mutating, accepting gene I_2 *in addition* to gene I_6 (not just in replacement), so that the limb module is now executed three times in three different regions instead of twice.

Structural Duplication and Divergence, or "Serial Homology" Later in the course of evolution, similar copies of the same organ can diverge and acquire specialized characteristics, as Fig. 11.10c illustrates. In this scenario, three copies of the entire limb module were produced by duplication as in Fig. 11.10b. Afterwards, these copies mutated independently from each other, e.g., by adopting different cell division rates p', which created shorter or longer limbs. *Serial homology* is the name given to this major evolutionary process resulting from duplication followed by divergence [7, 40]. Biological organisms often contain numerous repeated parts in their body plan. This is most striking in the segments of arthropods (several hundreds in millipedes; see the simulated "biomorphs" of [14]) or the vertebrae, teeth and digits of vertebrates. After duplication, these parts tend to diversify and evolve more specialized structures (lumbar vs. cervical vertebrae, canines vs. molars, etc.). Homology exists not only within individuals but also between different species, as classically shown by comparing the forelimbs of various tetrapods from the bat to the whale. Thus homology should also be explored as an important principle of artificial self-developing systems.

Structural Addition of Limbs by Body Plan Expansion In the scenario of Fig. 11.11d and e, new limbs are generated not by reusing the same body plan differently (Fig. 11.10a and b) or by duplicating the limb module (Fig. 11.10c), but rather by expanding the GRN of the base G_{PF} in order to create new regions of gene identity that can host additional limb growth. Here, the embryo's geography has expanded from a $3 \times 3 = 9$-type checkered pattern to a $5 \times 3 = 15$-type (Fig. 11.11d) and a $9 \times 3 = 27$-type pattern (Fig. 11.11e). The SA part of the body plan is also slightly modified to accommodate these new regions. It assumes an oval shape resulting from a *nonuniform* distribution of the division rate p that elongates the body along the Y axis (see Fig. 11.3), i.e., greater toward the N and S poles and lower in the middle.

Structural Addition of Digits by Modular Hierarchy Finally, along the same principles, Fig. 11.12 shows a few simulation trials of *three-tier* organisms. Figure 11.12a is the new wild type. After the usual development of two limbs from the 3×3 body plan, extra "digits" now grow from these limbs, guided by the top module of the hierarchical genotype. To make room for these digits, limbs have expanded their internal pattern from 1×1 to 2×4 (see previous paragraph). Figure 11.12a presents a double bilateral symmetry, with respect to both horizontal and vertical axes. Figure 11.12c is a further mutation of Fig. 11.12b, in which region I_6's limb has accelerated its growth and expanded its checkered pattern to support the development of two new digits, whereas, on the contrary, region I_4's limb has continued to regress back to a primitive "stump". Figure 11.13 paints a possible phylogenetic tree that includes all the species simulated in this section (dashed branches suggest "convergent" speciation pathways).

Naturally, beyond these proof-of-concept simulation trials, a more systematic exploration is needed. Further work needs to be done on how an embryomorphic system can *spontaneously* evolve, i.e., how it can be randomly varied and non-randomly selected based on its success in performing certain tasks. Toward this

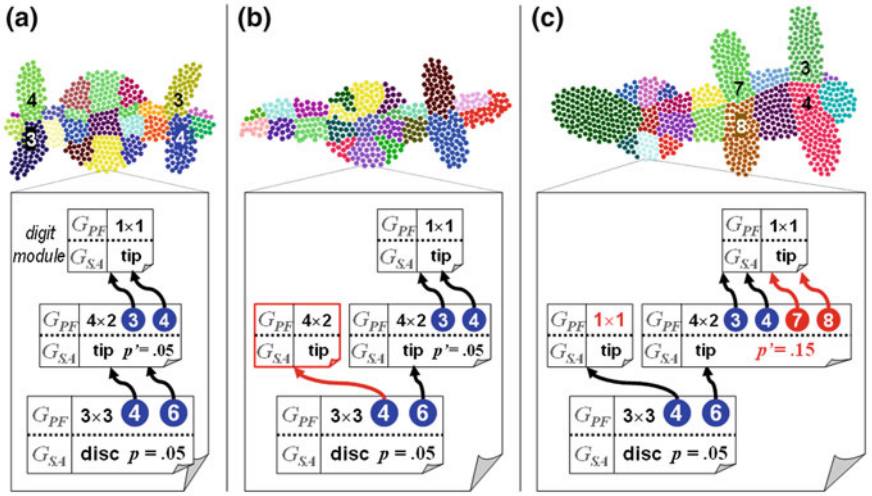


Fig. 11.12 **a–c** Adding digits to the limbs via a third tier in the modular hierarchy of the genotype (from [19])

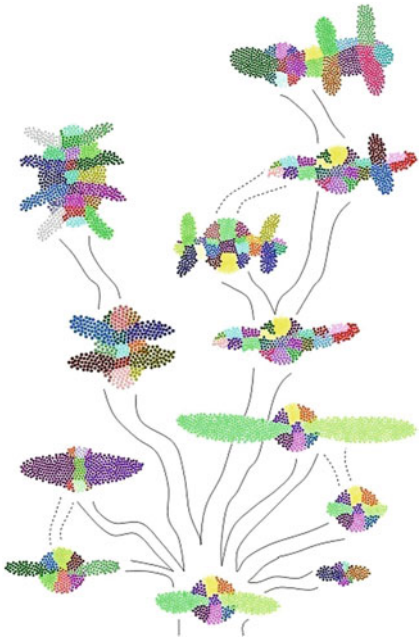


Fig. 11.13 A phylogenetic tree (from [19])

objective, different selection strategies are possible, whether focusing on prespecified forms, prespecified functions, or allowing open outcomes.

When selecting for **form**, a hard reverse engineering problem must be addressed: given a desired phenotype, what is the genotype that can produce it? While determin-

istic phenotype-to-genotype "compilation" is possible in limited cases [49], parameter search is generally difficult. With a fitness criterion rewarding only a specific target shape, solutions in genomic space are likely to be few and far between, if not reduced to a unique spot. In this situation, a classical approach is to define a "shape distance" as an increasing function of favorable, stepwise mutations. It is conjectured here that this kind of gradual search might actually benefit, not suffer, from the high genotype dimensionality of an embryomorphic model, compared to the direct genotype-phenotype mappings of most genetic algorithms. Hierarchical GRNs might be better at providing the fine-grained mutations required by the "gentle slope" search toward increasingly sophisticated innovation [14, 51]. Complex systems inherently have greater variational power, as they allow combinatorial tinkering on highly redundant parts.

However, beside gaining self-repair properties, why constrain a self-assembling system to produce a predefined shape? More benefits might come from such systems by selecting them for **function** while leaving complete freedom of form. Gradual optimization could rely on a distance of *performance* to predefined goals, instead of shapes, allowing the most successful candidates to reproduce faster and mutate. Functional selection under free form has often been tried in evolutionary robotic systems [41, 43], but there it was based on non-developmental, direct genotype-phenotype encodings. Again, it is hypothesized here that a larger number of microscopic agents, such as in multicellular embryogenesis, would be more favorable to a successful functional search due to their collective combinatorial abilities.

Finally, in a third scenario, specifications can be diversified and relaxed to the point of being **open** to surprise and harvesting unexpected but useful organisms from a "free-range menagerie" (see for example "evolutionary swarm chemistry" [57]). Ultimately, reconciling the antagonistic objectives of spontaneity and purpose will probably hinge on two complementary aspects: (a) finer-grained variation-by-mutation mechanisms yielding a larger number of search paths and (b) looser selection criteria yielding a larger number of fitness maxima. With more search paths covering more fit regions, evolution is more likely to find good matches.

11.4 Functional MapDevo by Animation in 3D

While the task of "meta-designing" laws of artificial development inspired from biology is already challenging, it only constitutes the first part of the Embryomorphic Engineering effort. Once a self-developing infrastructure is mature, what other computing and behavioral capabilities can it support? What do its "cells" (agents) and "organs" (regions) actually represent and achieve in practice? In biological organisms, although cell physiology often partakes in development (e.g., electrical signals of neurons guiding synaptogenesis), there seems to be a broad distinction between developmental genes and the rest of the genome. In computing systems, these two modes could also be separated into two different sets of state variables. After reaching developmental maturation, and while still fulfilling maintenance and self-repair

tasks, morphogenetic SA and PF activity (division, position information and patterning signals) would give way to another type of activity subserving functional computation. Obviously, the type of computation entirely depends on the nature of the agents: processor-carrying nano-units, software agents, robot parts, mini-robots, synthetic bacteria, and so on.

In fact, the problem is the opposite in many computing domains: there is a demand for precise self-formation capabilities in distributed systems made of existing functional agents. A variety of morphogenetic-like approaches have been proposed for such applications. For example, Amorphous Computing has set the stage for a myriad of micro-processors containing the same instructions to self-organize without an exact blueprint map or functional reliability, unlike traditional VLSI [1, 12, 49]. Self-assembling components can also represent mobile sensors and actuators in complex self-managing networks [3, 4]. In software applications (servers, security), a society of small-footprint software agents could diversify and self-deploy to achieve a desired level of application functionality and service (e.g., "immune" security [33]). It is also an important challenge in complex "techno-social networks" made of myriads of devices, software agents, and/or human users, which use only local rules and peer-to-peer communication to achieve a collective function [23] (see Sect. 11.5). In collective robotics, too, whether articulated parts of reconfigurable devices [29, 35, 41, 43], or mobile formations of mini-robots [9, 32, 70], there is a need for complex but controllable morphologies.

This section describes preliminary work toward such a goal through a model of animated MapDevo organisms immersed in a 3D physical environment, called *fMapDevo* [21]. The developmental process follows the exact same principles as the 2D model of Sect. 11.2 (SA by elastic forces, PF-I by gradient propagation, PF-II by gene expression). In addition, *after* development, mature organisms are able to generate movement by contracting adhesion links between "muscle" cells, while other cells have differentiated into "bones" and "joints" to support and articulate the body's structure. Finally, by interacting with a virtual physical world, made of a rigid floor, simple objects and a gravitational field, creatures can exhibit locomotion and primitive behavior. This project constitutes an original demonstration of a genuinely *evo-devo* Alife system, in which self-organization is not only programmable but functional and evolvable. We summarize below the main features and novelties of this project compared to MapDevo.

Body Growth in 3D Space We use the Open Dynamics Engine (ODE) to implement the embryomorphic development and behavioral dynamics of the organisms in 3D. Like the 2D version, cells are modeled as point-like elements (here represented by small spheres, Fig. 11.14) and neighborhood relationships are calculated by a 3D Delaunay triangulation (Fig. 11.14e) from which long links are removed above a cut-off distance. As before, mechanical SA forces are elastic links between neighboring cells, and in the first stage—the growth of the body—cell division is characterized by a uniform probability and random orientation (visualized with vectors, Fig. 11.14f). Gradient propagation PF-I (Fig. 11.14g) is triggered by three pairs of source cells, North-South, West-East, and Top-Bottom (Fig. 11.14a), which place themselves as

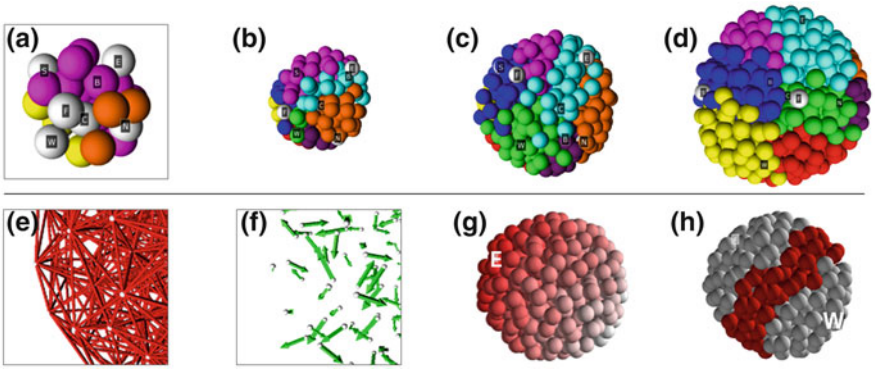


Fig. 11.14 Simulation of body growth in 3D space. **a** The 6 original source cells inside a small ball of cells after a few divisions at iteration $t = 25$. **b–d** Successive growth states at iterations $t = 150$, 400 and 700: 27 cell-type regions have formed under a $3 \times 3 \times 3$ checkered gene expression pattern. **e** Detail of the mesh of neighborhood links calculated by 3D Delaunay triangulation. **f** Detail of the random division vectors in each cell: norms represent probabilities, orientations are perpendicular to cleavage planes. **g** East gradient g_E from the E source, displayed in *red–white shading*. **h** A thick equatorial plane (in red) corresponding to $|g_W - g_E| \leq 3$

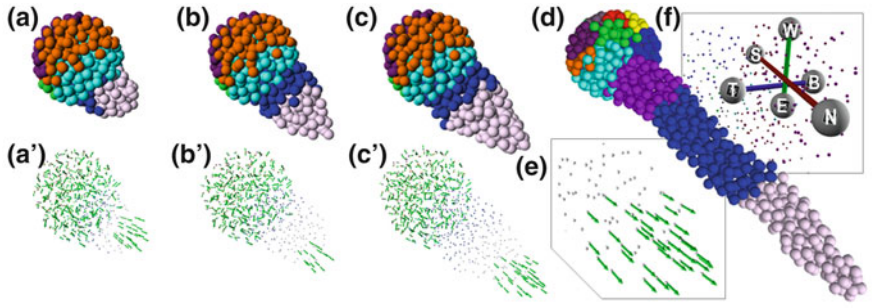


Fig. 11.15 Simulation of single limb development in 3D. **a–d** Successive states. **a′–c′** Corresponding division field: the body has halted its growth (nullifying all division vectors inside), while in the limb the division probability is 0 everywhere except at the local North tip ($g_N \leq 3$), where its orientation is South→North. **e** Detail of the division field at the tip. **f** The three pairs of self-positioned sources inside the limb (showing axes, not actual links), with South at its root and North at its tip

usual by hopping away from each other, i.e., navigating the opposite gradients uphill. Regional differentiation PF-II (Fig. 11.14b–d) results from the execution of a genetic program, whose output depends on the input gradients in each cell. In this model, however, the program is not necessarily a GRN but can be in symbolic format, such as logic rules (e.g., "if $|g_W - g_E| \leq 3$ then switch on the red gene", Fig. 11.14h).

Modular Limb Growth, Homology and Divergence In a second stage, limb growth (Fig. 11.15a–d) proceeds in the same way as the 2D version, by relying on a *heterogeneous* field of cell division probability and orientation (Fig. 11.15a′–c′), which

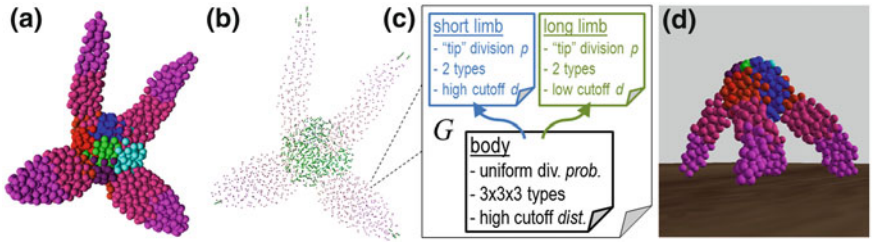


Fig. 11.16 Fully developed 4-legged organism. **a** Standard sphere-based multicellular view from underneath. **b** Corresponding division vector field, null in the limbs except at their tips. **c** Genetic program G executed by all cells, comprising three modules: a body module (uniform field of division probability, 27 cell types), a short-limb module (tip-area division field, 2 subtypes, high link cutoff), and a long-limb module (small link cutoff). Each limb module is triggered in two different regions of the body, creating a total of four legs. **d** Profile view of the creature when positioned on the floor

is calculated as a function of the local gradients inside the limb. In the example below, cell division is zero everywhere except at the North tip, where its orientation is South→North (Fig. 11.15e, f). As in the 2D model, the same "homologous" limb module of the genotype can be reused to develop several limbs from different "imaginal" regions of the body (Fig. 11.16). Then, evolutionarily "divergent" versions of that structure can be created by varying, for example, the link cutoff distance: a high value makes cells more likely to remain linked as neighbors, hence cluster together due to the elastic attraction and create more compact, shorter limbs. Conversely, a lower cutoff value tends to detach more cells from each other, hence let them spread out and make longer limbs. In Fig. 11.16, the developed organism possesses one pair of short limbs and one pair of long limbs. In sum, each module of the organism (body, limbs, etc.) represents an autonomous domain of space in which local gradients are mapped to various fields of developmental and structural parameters, such as division vectors, cell types, link cutoff value (and stiffness coefficient: see next), via a local genetic program (Fig. 11.16c).

Bones, Joints, Muscles: Structural Differentiation and Dynamics In the embryomorphic paradigm, the genotype-guided development of an organism not only provides a reproducible overall shape, but can also equip this shape with built-in structural features that confer it specific mechanical properties. In Figs. 11.16 and 11.17, for example, a few cells at the root of the limbs (where they attach to the body) have differentiated into "muscles", while others have become "bones" and "joints" inside the limbs. Computationally, this amounts to adding various Boolean fields—functions of the local gradients, like the division and type fields—to each genetic module (Fig. 11.16c). Here, the muscle field corresponds to a cylindrical section of the limb's root, e.g., where $g_S \leq 5$ (pink regions in Fig. 11.16a and d to be contrasted with the purple tips), while the bone field is 1 only along some thin South-North path on each limb and inside a small cluster at the center of the body. Link types are then simply deduced by connecting neighboring cells of identical types: for example, the bone links are formed exclusively between bone cells (white edges in

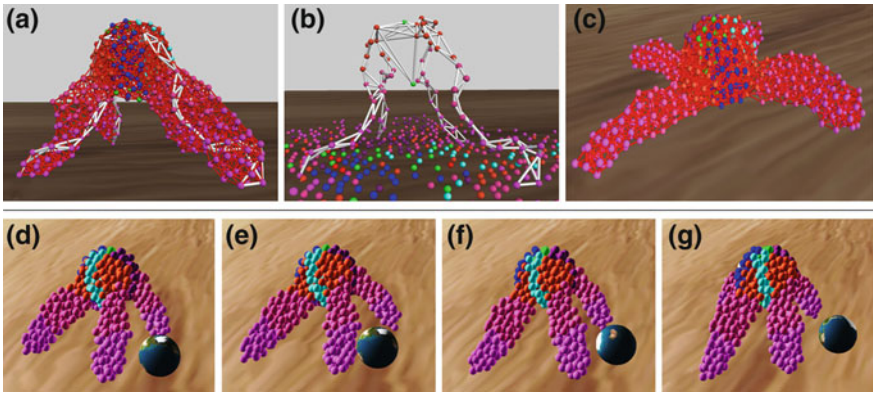


Fig. 11.17 Structural differentiation and dynamics. **a** The grown organism contains a skeleton made of differentiated "bone" cells and rigid links connecting them (displayed in *white*). **b** Experiment where all other links (the "flesh" in *red*) have been dissolved, showing the stability of the naked bone structure under gravitational pull. **c** Opposite experiment where bone differentiation was turned off: the organism spreads on the floor like a starfish. **d–g** Locomotion and ball-kicking behavior, achieved by stimulating and contracting the "muscle" regions (*pink* roots of the limbs) in specific subregions at specific time intervals—a coordination and control program that would be typically the task of a central nervous system

Fig. 11.17a, b). In this case, for a link to turn into "bone" means to become rigid, i.e., acquiring a virtually infinite spring coefficient, so that it maintains a fixed spatial relationship between its two extremities. The net effect is that a connected bone structure forms a "skeleton" that can support the whole organism and keep it standing on the floor under gravitational pull. The skeleton's stability can be revealed by "dissolving the flesh" around it as in Fig. 11.17b. Its usefulness can also be demonstrated by turning off bone differentiation, after which the softened organism collapses on the floor in a spread-out posture resembling a starfish (Fig. 11.17c).

Behavioral Performance and Evolution Finally and most importantly, once the mechanical features of cells and links have been established by development, the organism is immersed in a physical environment where it can exhibit locomotion and other types of behavior. In Fig. 11.17d–g, it is shown walking on the floor and kicking a ball. Without going into details here, this is essentially achieved by contracting the muscle regions (pink roots of the limbs) periodically and nonuniformly through "stimulus" fields applied to specific subregions at specific time intervals—a coordination and control program that would be typically the task of a central nervous system, itself subject to evolutionary changes. For more information on this model, the reader is referred to upcoming publications such as [21].

In sum, the fMapDevo model offers a complete *morphogenetic machine* that can transform by development a genotype G (Fig. 11.16c) into a functional phenotype (Fig. 11.16d). Metaphorically, G is the music roll of this mechanical organ, through which evolution can play different original tunes, i.e., produce different innovative architectures.

11.5 ProgNet: Programmable Network Growth

After the foundational 2D/3D embryomorphic models of the MapDevo family (Sects. 11.2–11.4), which remained close to their biological inspiration based on multicellular development, this part presents an extension to "*n*D" *graph topologies*. In this original project of programmable network self-construction and dynamics, called *ProgNet* (first published in [22]), neighborhood relationships between nodes are no longer necessarily a consequence of their proximity in Euclidean space. Yet, the overall challenge remains the same: design or evolve a ruleset that the individual agents of a multi-agent system can follow to independently create connections with each other, such that the end result is an intended functional architecture.

With information and communication technologies (ICT) pervading everyday objects and infrastructures, today's Internet, so far playing the role of a communication highway, is envisioned to become in the near future an "Internet of Things", i.e., a vast and hybrid complex network that will seamlessly integrate the physical and the virtual worlds. It will enable the spontaneous creation of collaborative societies of otherwise separate systems, both mobile and static, referred to as "cyber-physical ecosystems" (CPE) [64]. Examples will include self-reconfiguring manufacturing plants, self-stabilizing energy grids, self-deploying emergency taskforces [65], and self-growing autonomic applications [15]. What they will all have in common is a myriad of devices, software agents, and human users, dynamically building and reconfiguring their own network structures on the sole basis of local rules and peer-to-peer communication [23].

In this context, the ability to form specific connections by "programmed attachment" (as opposed to random connections by "preferential attachment" [2]) in a decentralized, self-organized way, would be of great benefit to a number of real-world situations where networking accuracy and reliability is important. Here, agents are called "nodes" and represent, for example, human users equipped with wireless devices such as personal digital assistants (PDAs), or software agents acting as proxies for physical machines and other resources that need to function together.

The basic mechanisms of self-constructing networks in ProgNet are explained in the following subsections from an abstract viewpoint. We start with elementary chains and continue with more complicated, composite architectures, including branching and stochastic redundancy. Nodes come in one by one and attach to the growing structure toward the goal of building a particular topology. They communicate with each other and execute the same program, but also gradually differentiate according to local and limited positional information in the form of discrete "gradients", similar to MapDevo. The self-assembly program carried by each node includes routines for the exchange of messages, the opening and closing of attachment ports and the dynamical creation or removal of links. Ports, gradients and other state variables guide new nodes to specific locations in the developing network. As the network expands and node positions change, nodes adapt by switching different rule-subsets on or off—analogous to gene promotion/repression in DNA—thus triggering the growth of specific structures.

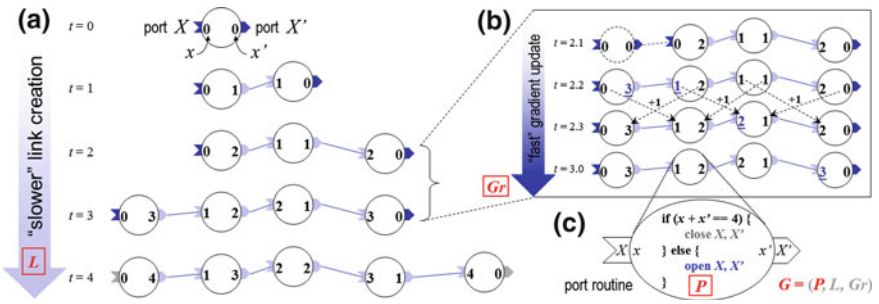


Fig. 11.18 Self-assembly of a simple chain. **a** The five main steps leading to a 5-node chain. Through the *link creation* routine L, incoming nodes attach to either open ports, X or X' (in *dark blue*), of the forming chain. When a link is created, its ports become "occupied" (in *light blue*) and gradient values are updated in all nodes (see **b**). When the chain length is 5, all open ports are closed (in *gray*; see **c**). **b** Detailed substeps of the value-passing *gradient update* routine Gr. **c** *Port management* routine P, the core and only evolvable component of genotype G in each agent: here, the ports of a node are instructed to close when $x + x' = 4$, i.e., length is 5 (adapted from [22])

Constructing Simple Chains Chains are the simplest self-assembling structures. In this first scenario, nodes possess two ports, X and X', and two corresponding gradient values x and x' (Fig. 11.18). Ports can be "occupied" (linked to other node ports) or "free" (not linked), while free ports can be "open" (available for a link) or "closed" (disabled). New nodes that just arrived in the system's space, or nodes that are not yet connected, have both ports open and gradients set to 0. A new node j can create a link with an existing node i only through a pair of complementary open ports, here X and X', with one link per port. Thus the only two possible links between i and j are $X'_j \leftrightarrow X_i$ or $X'_i \leftrightarrow X_j$. Upon attachment, gradient values are immediately updated according to the following rules: (a) a free port always maintains its value at 0 (gradient source), and (b) assuming that it was link $X'_i \leftrightarrow X_j$ that was created, value x is sent out in the direction $X'_i \rightarrow X_j$ with an increment of $+1$ so that $x_j = x_i + 1$, while x' is sent out in the opposite direction $X'_i \leftarrow X_j$ so that $x'_i = x'_j + 1$ (swap i and j if the other link was created instead). This is similar to the gradient propagation rule of the embryomorphic model presented in Sect. 11.2.

Figure 11.18 shows the self-assembly of a short chain. A new node can connect to any available open and complementary port at random, including the most recent and oldest nodes of the chain: all potentially valid links (here, two at any time) have an equal probability of being formed. The gradient counters keep track of the nodes' positions in the chain. This allows, for example, to build chains of a fixed length n by closing any remaining open ports as soon as $x + x' = n - 1$. Again, as mentioned in Sect. 11.2, discrete counter increments are also the method of choice for spreading positional information in other spatially extended paradigms [3, 4, 12, 49]. In the present model, the role of the gradient source can be transferred to another node, thereby shifting gradient chains in successive corrective waves, as nodes continually communicate with each other to adjust their counters. Figure 11.18b shows an

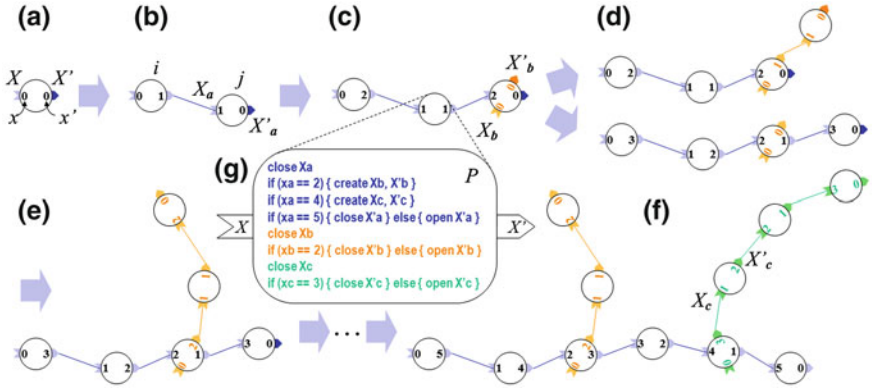


Fig. 11.19 Sketch of a programmed branching scenario. **a**, **b** Beginning of chain a (ports and links in *blue*). **c** Branch b starts to grow (*orange*). **d** Two alternative next steps. **e** Chain b stops growing at length 3. **f** Final developed structure, including a 4-node branch c (*green*). **g** This exact network is prescribed by the port management program P carried by each node (from [22])

example of a step-by-step decomposition of a gradient update after a new node has connected to the chain (dashed circle to the left).

In sum, all nodes carry the same program, genotype G, which comprises three main routines: *gradient update* (Gr), *port management* (P), and *link creation* (L):

- The gradient update routine, denoted Gr to distinguish it from G, was explained above: it consists of generic code that provides nodes with the positional information that they need to make further decisions, and is used in all network structures (see next sections).
- The port management routine P (Fig. 11.18c) contains the heart of the logic specific to the topology of a target architecture—chain, lattice, or any complicated composite graph. For example, in the case of a 5-node chain, P simply commands a node to shut its ports whenever $x + x' = 4$ (the "open" and "close" commands apply only to free ports, and are ignored by occupied ports).
- Finally, the link creation routine L (Fig. 11.18a) is also generic logic that prompts new nodes to pick one of the open ports of the network at random to make a new connection.

Routines Gr and P are executed only by the nodes that are already involved in the network, paving the way for newcomer nodes to execute routine L. In the remainder of the text, we focus on P, as it is the only variable and evolvable part of the genotype G (while Gr and L are stereotyped and fixed).

Branching and Modular Structures by Local Gradients More complicated structures can then develop by composing multiple chains and lattices. To allow the creation of *modules* with their own identities and local positional information, one can find again inspiration from biology, in particular the concepts of *modularity* and

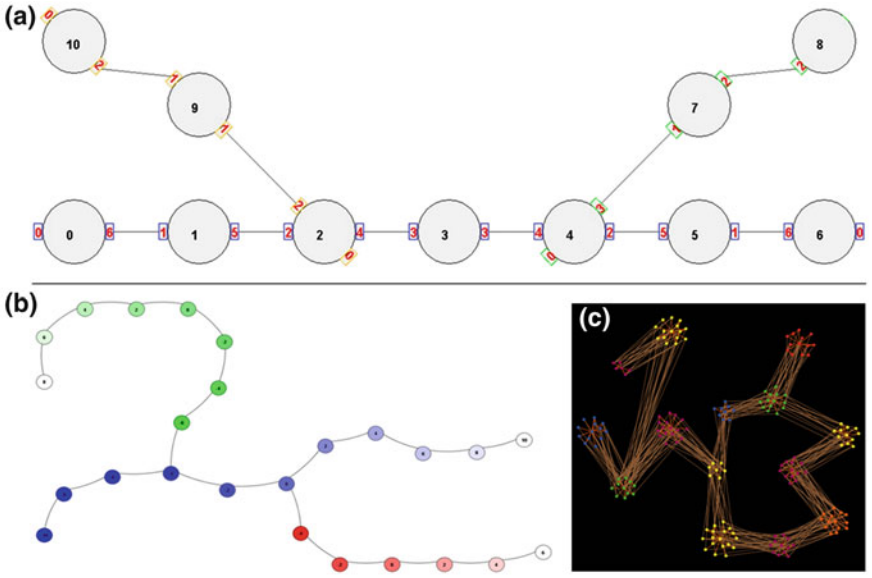


Fig. 11.20 Simulation trials of programmed branching structures. **a** A main chain (*horizontal*, here) branches off into two smaller chains at points where the x gradient values respectively reach 2 and 4. Nodes show here a unique ID number, which is *not* playing any role in the attachment dynamics, while ports (represented by *small rectangles* attached to the nodes) contain the gradient values. Every node carries all three pairs of ports (*blue*, *orange*, *green*) but uses only 1, 2 or 3 single ports (resp. at chain extremities, middles, and junctions). **b** Another example of chain (in *blue*) branching off into a *red chain* at $x = 3$ and a *green chain* at $x = 5$. Here, the layout follows a force-based algorithm and integer gradient values are visualized by color shading (from [20]). **c** Example of a complex programmed network integrating a branching chain structure with cluster formations (from [22])

homology that are central in evo-devo [8, 40, 48] (see Sect. 11.3). Modules are similar to "limbs" that have distinct morphologies and geographies. They are implemented here by distinguishing between chain segments with independent coordinate systems based on different "tags" a, b, c, etc. To start with a simple example, a new chain can branch off from the middle of another chain (Fig. 11.19). The gradient ports in the initial chain of the system are denoted by (X_a, X'_a), while the ports of the branches will be (X_b, X'_b), (X_c, X'_c), etc. Accordingly, routine L is modified so that links cannot be created between ports with different tags.

In the elementary scenario of Fig. 11.19, when the third node has attached (i.e., when $x_a = 2$), the P routine commands that a new pair of ports (X_b, X'_b) be created on that node and only port X'_b be opened (Fig. 11.19c). Afterwards, new nodes can attach to either open port, X'_a (lengthening the initial chain) or X'_b (starting the new branch; Fig. 11.19d). Under the right set of constraints, generally imposing unidirectional attachment (e.g., always to X'), the order of node attachment does not influence the final structure. Actual simulation trials of self-organized branching

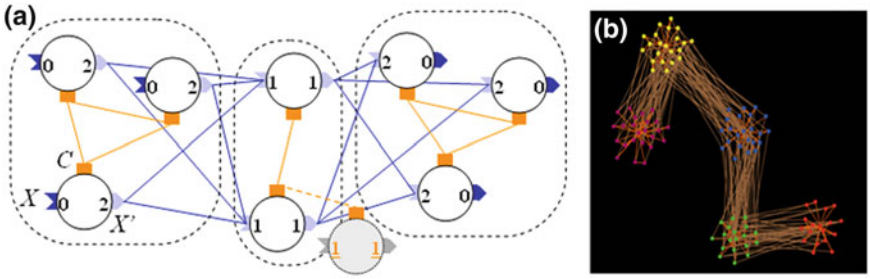


Fig. 11.21 Chain of node clusters. **a** Detailed 3-cluster chain: internal (*orange*) links connect the C ports of nodes with same (x, x') values, while (*blue*) links between clusters form the chain. A new node (*gray*) connecting through C adopts the cluster's values. **b** Simulation with 5 clusters and about 20 nodes per cluster (from [22])

structures are shown in Fig. 11.20, including a composite structure in which nodes are "thickened" into clusters by adding a C port, as explained below (Fig. 11.21).

Robustness by Cluster Redundancy The previous examples involved exact structures of connections that were programmed at node-level by a (quasi) deterministic algorithm. Despite minimal randomness in the choice of locations for new attachments, there was a unique possible final outcome: a chain or a branched structure planned in advance. While we want to preserve this essential property of *programmability* (the focus of this work and of Morphogenetic Engineering in general), it is also important to reintroduce an element of *variability* and *redundancy* in the system—albeit at a smaller scale. In biological development, the position and number of individual cells is very imprecise, while the tissues and organs they form are reliably placed. Similarly, programmed network self-assembly can also afford to be irregular at the microscopic level of the nodes, while retaining an orderly arrangement at the higher, mesoscopic levels of groups of nodes.

One way to implement this idea is to simply "thicken" chains and lattices (Fig. 11.21) by replacing single nodes with *clusters* of nodes. This can be done through one additional port, C (as in "cluster" or "clique") that allows multiple nodes with identical gradient coordinates to form random connections with each other. In chains, the C port represents an extra "nonlinear" dimension on top of the (X, X') pairs of ports. Another new feature is that nodes are now allowed to make multiple connections per port, whether X, X' or C (Fig. 11.21a). As a result, nodes tend to cluster into families according to their gradient values. Thus a new node generally faces two types of attachment possibilities: it can either thicken or lengthen the chain. Similar to cellular proliferation in morphogenetic tissues and organs, this proliferation of nodes *within a structured network* introduces redundancy and "failover" safety. Overall, however, it remains a deterministic structure (guided by the genotype of attachment rules P) on top of fine-grained stochasticity.

Adaptive Growth In sum, ProgNet proposes abstract principles of self-made networks capable of forming precise topologies in a purely endogenous manner. It

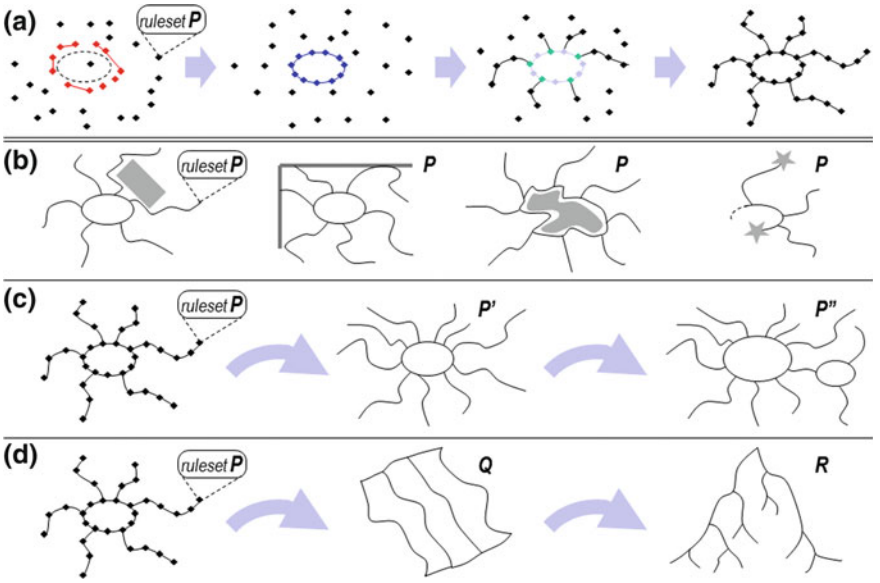


Fig. 11.22 Illustration of various types of phenotypic adaptation in a programmable network growth model. **a** Stereotyped development: a certain genotype (port routine P) gives nodes a strong bias toward self-assembling into a certain shape, here a spider-like formation made of one ring and six legs. **b** Developmental "polyphenism": similar to a plant, the same P could give rise to variants of the above shape modified by external conditions from the environment, such as obstacles or attractors. **c** "Polymorphism": slight parametric variants of P may produce other structural variants, such as size of ring, number of legs, or ring location. **d** "Speciation": drastically different genomes create drastically different structures—although there is no real qualitative difference with **c**: it is only a matter of degree and time scale of evolution

establishes generic rules for the emergence of non-random (except for possible redundancies at the microscopic level), programmable graph structures that are neither repetitive nor imposed by external conditions. Beyond the engineering of stereotypical genotype-phenotype mappings, however, network growth must also be *adaptive*. It is critical to be able to rely on dynamic structures that can co-develop with a rapidly changing situation by remaining open to influences and modifications coming from the environment in which they are expected to function (Fig. 11.22). This can occur on multiple taxonomic levels: on the long time scale through *speciation* reflecting "new" genotypes (Fig. 11.22d), on the shorter time scale through *polymorphism* of a "single" species (Fig. 11.22c), or even on one individual's time scale through developmental *polyphenism* (Fig. 11.22b):

- **Evolutionary Polymorphism: Varying the Genotype** A genotype may provide internal parameters controlling different "traits" of the final structure: slight variants of the former produce slight variants of the latter (Fig. 11.22c). This is similar to the classical laws of population genetics *within* the same species, schematically corresponding to the concepts of "alleles" or single-nucleotide polymorphisms

(SNPs) in DNA. Varying and combining genotypic parameters gives rise to a family of different "breeds"—like Mendel's peas or Darwin's pigeons. Note, however, that the distinction between polymorphism and speciation (Fig. 11.22d) is not clear cut: it is only a matter of degree and time, as the same evolutionary mechanisms are at work in both cases.

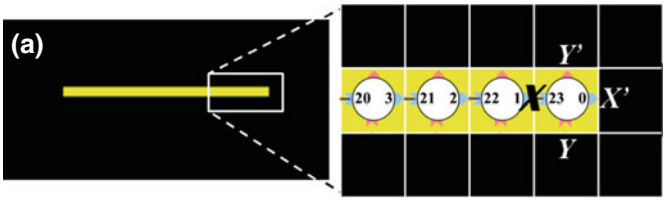
- **Developmental Polyphenism: Varying the Phenotype** Under an invariant genotype, however, development can also be modified by environmental conditions (Fig. 11.22b). External cues surrounding one individual during its growth can also play an important role in its final structure. This is the level of the phenotype, for which natural analogies can be found more readily in the vegetal kingdom: plants and trees can be pruned, bent, arranged, or sculpted, whether by human intervention (bonsais, espaliers, topiaries, etc.) or by natural conditions (wind, rocks, soil, light, etc.).

11.6 ProgLim: Program-Limited Aggregation

A number of real-world networks combine non-spatial graph topologies (e.g., connecting software agents or organizations) with Euclidean graph topologies (e.g., connecting people and equipment on the field) at different degrees. For example, many cyber-physical systems inherently have a dual spatial/non-spatial nature, as they often include a physical infrastructure at a lower communication level, together with a virtual overlay network at a higher application level [65]. The abstract mechanisms of programmed attachment in the above ProgNet framework create purely non-spatial graphs, which can still be viewed in 2D by using a force-based layout algorithm [27]. But if nodes represent agents and devices interacting in real space, the dynamics, not just the visualization, should also be modified to take into account the effects of metric distance on node aggregation.

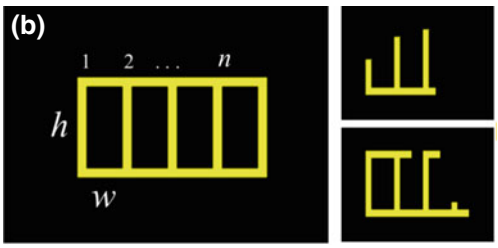
In the particular embodiment of ProgNet presented here, called *ProgLim* (for "Program-Limited Aggregation"), we revert to the 2D plane and restrict nodes to discrete positions on a grid. By simplifying the network's space, we can gain better control and understanding of its embryomorphic dynamics. Here, each node can have at most four neighbors, and create up to two horizontal links, left and right, and two vertical links, up and down. They are the equivalent of square pixels in a 2D cellular automaton (displayed in yellow on a black background in the figures below), whose four ports X, X', Y and Y' are located at the centers of their four edges (Fig. 11.23a). As before, incoming nodes aggregate to the structure one at a time by choosing any currently free edge at random. The next subsections give a brief overview of ProgLim, which includes preliminary experiments combining evolution and polyphenism (for more details, see upcoming publications such as [20]).

Acquiring Polyphenism by Evolution In ProgNet, node attachment was only based on port availability driven by positional gradient values: a network grew in vacuum, while environment-induced polyphenism remained theoretical (Fig. 11.22b).



Port Routine ***P*1** : Line

open X, X'
close Y, Y'



Port Routine ***P* 2** : Regular Row of Rectangles

$h = 10, w = 5, n = 4$
if $(x' = 0)$ **then** open X'
if $(x \,\%\, w = 0)$ **then** open Y'
if $(y > 0)$ **then** close X'
if $(y = h)$ **then** open X'
if $(x \geq nw)$ **then** close X'
if $(y \geq h)$ **then** close Y'

Fig. 11.23 Two simple stereotyped network examples on a 2D grid. All structures are made of *yellow square* nodes. **a** Open-ended line: the corresponding genotype (port routine P_1) simply consists of two unconditional port-opening actions, left and right, keeping the bottom and top ports closed. **b** Row of adjacent rectangles, growing toward the right and the top (with two intermediate stages shown in *inset*): in this case, genotype P_2 is more complicated, as it involves opening and closing the right and top ports (X' and Y') under certain conditions based on the gradient states and three parameters: w for the width (number of pixels) of each rectangle, h for their height, and n for their total number

In ProgLim, we can more easily experiment with the ability of the growth dynamics to be perturbed and diverted by obstacles—here, taking the form of "rocks" randomly scattered on the grid (Fig. 11.24). In practice, this is achieved by inserting pixel-state conditions in the port-opening rules, in addition to gradient-state conditions. Generally, in an empty (fully black) environment, the same genotype (port routine P) reliably creates the same network. In a cluttered environment, however, rocks (gray pixels) can block ports and impede growth. This is why variants of the genotype that are able to literally "work around" those obstacles and create networks similar to a desired wild type can be very useful. Contrary to an inflexible ruleset P, an adaptive ruleset Q can continue development in restrictive environments by providing bifurcations based on *neighboring* pixel states in the port-opening logic. As explained below, "rock sensing" is purely local, i.e., pixel-based influences only involve the states of the four nearest neighbors.

With the goal of finding adaptive genotypes Q, instead of designing them by hand, we apply an evolutionary algorithm to P. For this, we need to define a target structure that the network should ideally realize while at the same time dealing with obstacles. Precisely because of environmental perturbations, it will not reproduce the exact same configuration (especially on a discrete 2D grid). Yet, certain criteria can be designed to come as close as possible to the initially intended network. We demonstrate this principle below on two simple structures: an open-ended line formation (Fig. 11.23a)

and a row of adjacent rectangles (Fig. 11.23b). These two examples are especially interesting because they illustrate two different goals: a line can be construed as a tool to discover the environment in a particular direction, while a row of rectangles can be construed as a case of *modular* self-organization.

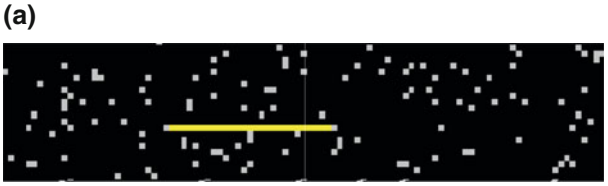
Rulesets and Mutations To let structures evolve and find good solutions, rulesets P are represented in standardized format using a grammar, and a list of possible mutation operators are defined. In short, each rule is written "**if** (*clause*1 [**and**|**or**] *clause*2) **then** *action*", where *clause*1 is based on gradient states only, *clause*2 is based on neighboring pixel states only (i.e., whether specific ports of the central pixel are hindered or not), and *action* manages one of the four ports as follows: "[open|close] $[X|X'|Y|Y']$". Each clause can be replaced by Boolean constants "true" or "false". Five types of mutations are considered: (i) inserting a random rule (possibly with a new constant value), (ii) deleting a rule, (iii) modifying a component of a rule (*clause* 1, *clause*2, [and|or], *action*), (iv) reordering a rule (switching its rank in the priority list), and (v) changing a constant (in the rectangle example: w, h, or n).

Fitness and Evolutionary Algorithm The goal function or "fitness" reflects the overall structure that we want to achieve:

- In the example of the open-ended line, the fitness is equal to L^2/N, where L is the horizontal extension of the chain (which might be less than the number of nodes if the line is diagonal, see below) and N is the total number of nodes. The intention is that the chain should stretch out as much as possible in one preferential direction without twisting and turning.
- In the row of rectangles, the fitness is the number of completed compartments, i.e., for which a closed border (possibly irregular) can be detected.

For a start, we use a primitive "(1 + 1)" evolutionary algorithm, i.e., not based on a population but on a single individual. At every time step t, one of the five mutation operators (i)-(v) is applied at random to the current ruleset P_t, generating a new ruleset P'. If the fitness of the new structure developed from P' is higher than the fitness of the structure developed from P, then $P_{t+1} = P'$; otherwise, $P_{t+1} = P$ with probability $1 - p$, or P' with probability p, where p is a probability of accepting a lesser fitness and varies as $1/\log(t)$ (a classical stochastic scheme akin to the "Monte Carlo" or "simulated annealing" methods, which can avoid being stuck in local optima).

Different numbers of trials per mutation and numbers of time steps necessary to find a good ruleset have been tested (discussed below). However, many mutated rulesets led to potentially infinitely growing networks, therefore we also imposed a global maximal number of nodes $N_{\max}$ at which development stopped. This corresponds to a situation of "limited resources" keeping swarms small in practice. This change has important consequences when $N_{\max}$ is lower than the total number of nodes N necessary to build a complete structure. In that case, the network ends in some arbitrary intermediate stage that depends on the order of node aggregation—although the final structure is often deterministic and, ultimately, should not depend on that order.



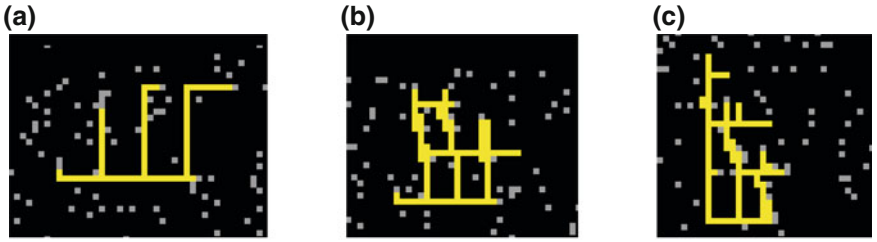
Por Routine *P*3: Fixed Line (like P_1)

open X, X'
close Y, Y'

Port Routine *P*4: Polyphenic Line

open X
if $(r(X) = 1)$ open Y'
if $(r(X') = 1)$ close Y'

Fig. 11.24 Evolution of the fixed line into a polyphenic line. **a** The same ruleset as P_1, this time in an environment littered with "rocks" (*gray pixels*), produces a straight line whose growth is rapidly blocked at both extremities. **b** After a few dozen mutations and selection steps, one of the evolved rulesets, P_4, is able to unblock the line growth (toward the left) by opening the top port Y' whenever a rock is encountered by the left port X



Port Routine *P*5 : Rectangles (b)

$h = 8, w = 5, n = 4, c = 9$
if $(y \,\%\, h = 0)$ **then** open X'
if $(x \,\%\, w = 0)$ **then** open Y'
if $(y = h)$ **then** close Y'
if $(x = nw)$ **then** close X'
if $(r(Y') = 1)$ **then** open X
if $(y' > c \;\&\; r(X') = 1)$ **then** open X

Port Routine *P*6 : Rectangles (c)

$h = 8, w = 5$
if $(y \,\%\, h = 0)$ **then** open X'
if $(x \,\%\, w = 0)$ **then** open Y'
if $(r(X) = 1)$ **then** open Y'
if $(r(X') = 1)$ **then** open X

Fig. 11.25 Evolution of the row of rectangles. **a** The development of the original ruleset P_2 is blocked by every rock on its way. **b** After a few hundred selected mutations, the structure can bypass the obstacles in certain directions and reform irregular compartments. The evolved ruleset P_5 is also relatively simplified compared to P_2. **c** Another 100 mutations later, the structure is able to grow farther out under an even more reduced ruleset P_6

Results As expected, the (1 + 1) evolutionary algorithm does not easily produce good solutions: the majority of mutations are deleterious or neutral, bringing the structure in a domain of genomic space where most "neighboring" genomes (one mutation away) have a low fitness. This happens usually because an action critical to the successful growth of the structure (e.g., "open X'" in Fig. 11.23) was accidentally

deleted from the ruleset, making it very difficult to recover that specific action through a reverse accident. Regardless, evolution is still very much possible, and even in this simple evolutionary framework, a number of decisive breakthroughs were observed:

- After a few dozen mutations, the fixed line's ruleset of Fig. 11.24a (the same as Fig. 11.23a in a cluttered space) has transformed into the three rules of Fig. 11.24b, which give the chain formation the necessary flexibility to bypass obstacles. In this case, it grows only toward the left by aggregating pixels to port X (first rule) but also to the top port Y', which is opened whenever a rock is encountered on its path (second rule, in which $r(X) = 1$ means "left pixel is gray"). Interestingly, the third rule is useless because, due to the direction of growth, there will never be a rock on the right-port side X'. Yet, that rule was resistant to further mutation or deletion for over 200 time steps, eventually remaining neutral.
- Concerning the row of rectangles, just as with the line, every rock blocks the initial ruleset's growth and the structure cannot be completed (Fig. 11.25a). After hundreds of selected mutations, however, the algorithm manages to grow past some obstacles and form new compartments (Fig. 11.25b). Interestingly, its interaction with the environment is not very complex (only the last two rules contain pixel-state conditions, plus a new constant) while the environment-independent part has been reduced to four rules, which are very different from the original. Another observation is that the constants did not evolve much, probably because in a random environment like this one there are no regularities that could be exploited to grow faster. After a hundred more mutations, the algorithm is able to surmount almost every obstacle (Fig. 11.25c). The ruleset has become even simpler, in particular the two previous terminating rules (third and fourth) have disappeared. The evolved structure is now relying on the rocks themselves and the limited total number of nodes to stop its own extension.

These preliminary experiments demonstrate that effective mutated genotypes can be rather short, even shorter than the original wild-type ones. We can also note that among the first rules to disappear during evolution were the ones closing ports and limiting growth in certain directions. In a cluttered environment, indeed, such rules are no longer needed as the rocks themselves can provide the necessary spatial frame. Moreover, the best rulesets usually contain rules that depend either on the gradients or on the environment but not both conditions at the same time. In fact, when new rules (brought by the insertion operator) were restricted to be exclusively environment-dependent, the fitness increased faster. Rules can also become "neutral" (as the third rule of P_4) in the sense that they never apply to any node in practice, thus do not interfere with the structure's growth. However, as it is generally well known in evolutionary computation methods, neutral elements also constitute an important reservoir of future useful mutations. Finally, the above evolutionary algorithm can obviously be improved in many ways (by screening mutations, by including a full-fledged population dynamics and crossover between individuals, etc.) to refine the search and obtain optimal rules while staying closer to the originally intended shape.

In summary, by reframing ProgNet within a regular 2D grid, the ProgLim project makes it easier to highlight the evolutionary potential of self-organizing program-

mable networks. In particular, it shows that stereotyped and brittle embryomorphic processes (Fig. 11.22a) can evolve to become more robust and *polyphenic* (Fig. 11.22b) under environmental pressure.

11.7 Conclusion

Embryomorphic Engineering is inherently interdisciplinary, as it closely follows biological principles at an abstract level, but does not attempt to model detailed data from real genomes or organisms. Thus, it sits at crossroads between different domains, from developmental and systems biology to artificial life, in particular spatial computing, evolutionary programming and swarm robotics. Following the tenets of Morphogenetic Engineering described in this book, it constitutes an original attempt to "endow a physical system with information" or, from the opposite viewpoint, "embed an informational system in physics" (see Introduction chapter). It does so by combining (1) mechanical self-assembly (SA) and (2) computational pattern formation (PF), (3) under the control of a genomic program (G):

- In MapDevo, these principles are modeled by dynamical processes, respectively: (1) cell adhesion (through elastic forces), (2) morphogen diffusion (through integer counters), and (3) gene expression (through a GRN).
- In ProgNet and ProgLim, they take the form of logical instructions, respectively: (1) the link attachment routine L, (2) the gradient propagation routine Gr and (3) the port-opening routine P.

Only few previous theoretical models of biological development or bio-inspired artificial life systems have combined these principles in various ways. The evo-devo works of [25, 34, 54], or with lesser morphogenetic abilities those of [49, 60], are among these early notable achievements. Other interesting studies have explored the combination of two out of three:

- SA and PF, no G: self-assembly by cell adhesion and signal-based pattern formation, but using predefined cell types without internal genetic variables [44].
- PF and G, no SA: non-trivial pattern formation by information-driven signaling, but on a fixed lattice without self-assembling motion [12, 13].
- SA and G, no PF: heterogeneous swarms of genetically programmed, self-assembling agents, but in empty space without mutual exchange of differentiation signals [55, 56].

More recently, new models of gene-controlled animats based on body-brain co-evolution and co-development have also shown a promising path toward a fully integrated artificial evo-devo approach [36, 37, 59].

Ultimately, abstracting farther away from biological development, an important goal of Embryomorphic Engineering is to contribute to the design of new self-organizing systems able to replace omniscient architects with large-scale decentralized collectivities of agents—the whole topic of this book. Many research works have

investigated the possibility of obtaining self-formation properties from a variety of complex computing components: nano-units, bacteria, software agents, robot parts, mini-robots, and so on (see the other chapters). Since functionality is distributed over a great number of components, it would be an insurmountable task to assemble and instruct each of them individually. Rather, in a way similar to biological cells, these components should be easily mass-produced, initially as identical copies of each other, and only acquire their specialized positions and functions by themselves within the system, once mixed together.

Acknowledgments Since the inception of Embryomorphic Engineering in 2006, R. Doursat's positions have been funded by the Brain Lab and Department of Computer Science, University of Nevada, Reno; the Complex Systems Institute, Paris Ile-de-France (ISC-PIF), CNRS; and the Research Group in Biomimetics (GEB), Universidad de Málaga, Spain. C.A. Sánchez is a PhD student at GEB since 2011. R. Dordea and D. Fourquet are MSc students by Ecole Polytechnique, Paris. T. Kowaliw is a research scientist at ISC-PIF since 2010, supported by Région Ile-de-France and the French ANR project grant "SynBioTIC" 2010-BLAN-0307-03.

References

1. Abelson, H., Allen, D., Coore, D., Hanson, C., Homsy, G., Nagpal, R., Rauch, E., Sussman, G.J., Weiss, R.: Amorphous computing. Commun. ACM **43**(5), 74–82 (2000)
2. Barabási, A.L., Albert, R.: Emergence of scaling in random networks. Science **286**(5439), 509–512 (1999)
3. Beal, J., Bachrach, J.: Infrastructure for engineered emergence on sensor/actuator networks. IEEE Intell. Syst. **21**(2), 10–19 (2006)
4. Beal, J., Dulman, S., Usbeck, K., Viroli, M., Correll, N.: Organizing the aggregate: languages for spatial computing. Comput. Res. Repos. **abs/1202.5509** (2012)
5. Bentley, P., Kumar, S.: Three ways to grow designs: a comparison of embryogenies for an evolutionary design problem. In: Proceedings of the Genetic and Evolutionary Computation Conference, vol. 1, pp. 35–43. Morgan Kaufmann, San Francisco (1999)
6. Callebaut, W., Rasskin-Gutman, D.: Modularity: understanding the development and evolution of natural complex systems. The MIT Press, Cambridge (2005)
7. Carroll, S.B.: Endless Forms Most Beautiful: The New Science of Evo Devo and the Making of the Animal Kingdom. W. W. Norton, New York (2005)
8. Carroll, S.B., Grenier, J.K., Weatherbee, S.D.: From DNA to Diversity: Molecular Genetics and the Evolution of Animal Design. Wiley-Blackwell, Malden (2001)
9. Christensen, A.L., O'Grady, R., Dorigo, M.: Morphology control in a multirobot system. IEEE Robot. Autom. Mag. **14**(4), 18–25 (2007)
10. Coen, E.: The Art of Genes. Oxford University Press, Oxford (2000)
11. Coen, E., Rolland-Lagan, A.G., Matthews, M., Bangham, J.A., Prusinkiewicz, P.: The genetics of geometry. Proc. Natl. Acad. Sci. U. S. A. **101**(14), 4728–4735 (2004)
12. Coore, D.N.: Botanical computing: a developmental approach to generating interconnect topologies on an amorphous computer. Ph.D. thesis, MIT (1999)
13. von Dassow, G., Meir, E., Munro, E.M., Odell, G.M.: The segment polarity network is a robust developmental module. Nature **406**, 188–192 (2000)
14. Dawkins, R.: Climbing Mount Improbable. W.W. Norton & Company, New York (1996)
15. Diaconescu, A., Lalanda, P.: A decentralized, architecture-based framework for self-growing applications. In: Proceedings of the 6th International Conference on Autonomic Computing, pp. 55–56. ACM (2009)

16. Doursat, R.: The growing canavas of biological development: multiscale pattern generation on an expanding lattice of gene regulatory networks. Inter J. Complex Syst. **1809** (2006)
17. Doursat, R.: Organically grown architectures: creating decentralized, autonomous systems by embryomorphic engineering. In: R.P. Würtz (ed.) Organic Computing, Understanding Complex Systems, pp. 167–199. Springer, Heidelberg (2008)
18. Doursat, R.: Programmable architectures that are complex and self-organized: from morphogenesis to engineering. In: Artificial Life XI: Proceedings of the 11th International Conference on the Simulation and Synthesis of Living Systems (Alife XI), pp. 181–188. MIT Press, Cambridge (2008)
19. Doursat, R.: Facilitating evolutionary innovation by developmental modularity and variability. In: Proceedings of the 11th Annual Conference on Genetic and Evolutionary Computation (GECCO), pp. 683–690. ACM (2009)
20. Doursat, R., Fourquet, D., Dordea, R., Kowaliw, T.: Morphogenetic engineering by program-limited aggregation. To appear (2013).
21. Doursat, R., Sánchez, C., Fernández, J.D., Kowaliw, T., Vico, F.: Function from structure from development: a dynamical evo-devo model of complex artificial organisms. To appear (2013).
22. Doursat, R., Ulieru, M.: Emergent engineering for the management of complex situations. In: Proceedings of the 2nd International Conference on Autonomic Computing and Communication Systems, vol 14. ICST, Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering, 2008
23. Dressler, F.: Self-Organization in Sensor and Actor Networks. Wiley, New York (2007)
24. Edelman, G.M.: Topobiology: An Introduction to Molecular Embryology. Basic Books, New York (1988)
25. Eggenberger, P.: Evolving morphologies of simulated 3d organisms based on differential gene expression. In: Proceedings of the Fourth European Conference on, Artificial Life, pp. 205–213, 1997
26. Floreano, D., Mattiussi, C.: Bio-Inspired Artificial Intelligence: Theories, Methods, and Technologies. The MIT Press, Cambridge (2008)
27. Fruchterman, T.M.J., Reingold, E.M.: Graph drawing by force-directed placement. Softw. Pract. Exp. **21**(11), 1129–1164 (1991)
28. Gierer, A., Meinhardt, H.: A theory of biological pattern formation. Biol. Cybern. **12**(1), 30–39 (1972)
29. Goldstein, S.C., Campbell, J.D., Mowry, T.C.: Programmable matter. Computer **38**(6), 99–101 (2005)
30. Goodwin, B.C.: How the leopard changed its spots: the evolution of complexity. Scribner, New York (1994)
31. Grégoire, G., Chaté, H.: Onset of collective and cohesive motion. Phys. Rev. Lett. **92**(2), 025702–025704 (2004)
32. Groß, R., Bonani, M., Mondada, F., Dorigo, M.: Autonomous self-assembly in swarm-bots. IEEE Trans. Robot. **22**(6), 1115–1130 (2006)
33. Hofmeyr, S.A., Forrest, S.: Architecture for an artificial immune system. Evol. Comput. **8**(4), 443–473 (2000)
34. Hogeweg, P.: Evolving mechanisms of morphogenesis: on the interplay between differential adhesion and cell differentiation. J. Theor. Biol. **203**(4), 317–333 (2000)
35. Hornby, G.S., Pollack, J.B.: Creating high-level components with a generative representation for body-brain evolution. Artif. Life **8**(3), 223–246 (2002)
36. Joachimczak, M., Kowaliw, T., Doursat, R., Wrobel, B.: Brainless bodies: controlling the development and behavior of multicellular animats by gene regulation and diffusive signals. In: Artificial Life 13: Proceedings of the Thirteenth International Conference on the Simulation and Synthesis of Living Systems, pp. 349–356, 2012
37. Joachimczak, M., Wróbel, B.: Evo-devo in silico: a model of a gene network regulating multicellular development in 3d space with artificial physics. In: Artificial Life XI: Proceedings of the Eleventh International Conference on the Simulation and Synthesis of Living Systems, pp. 297–304, 2008

38. Kauffman, S.A.: The Origins of Order: Self Organization and Selection in Evolution. Oxford University Press, Oxford (1993)
39. Kauffman, S.A.: Reinventing the Sacred: A New View of Science, Reason, and Religion. Basic Books, New York (2008)
40. Kirschner, M.W., Gerhart, J.C.: The Plausibility of Life: Resolving Darwin's Dilemma. Yale University Press, New Haven (2005)
41. Komosinski, M., Rotaru-Varga, A.: Comparison of different genotype encodings for simulated three-dimensional agents. Artif. Life **7**(4), 395–418 (2001)
42. Kondo, S., Asai, R.: A reaction-diffusion wave on the skin of the marine angelfish pomacanthus. Nature **376**, 765–768 (1995)
43. Lipson, H., Pollack, J.B.: Automatic design and manufacture of robotic lifeforms. Nature **406**(6799), 974–978 (2000)
44. Marée, A.F.M., Hogeweg, P.: How amoeboids self-organize into a fruiting body: multicellular coordination in dictyostelium discoideum. Proc. Natl. Acad. Sci. U. S. A. **98**(7), 3879–3883 (2001)
45. Meinhardt, H.: The Algorithmic Beauty of Sea Shells. Springer, Berlin (2003)
46. Miller, J.F., Banzhaf, W.: Evolving the program for a cell: from french flags to boolean circuits. In: On Growth, Form and Computers, pp. 278–301, 2003
47. Mjolsness, E., Sharp, D.H., Reinitz, J.: A connectionist model of development. J. Theor. Biol. **152**(4), 429–453 (1991)
48. Müller, G.B., Newman, S.A.: Origination of Organismal Form: Beyond the Gene in Developmental and Evolutionary Biology. The MIT Press, Cambridge (2003)
49. Nagpal, R.: Programmable self-assembly using biologically-inspired multiagent control. In: Proceedings of the First International Joint Conference on Autonomous Agents and Multiagent Systems: Part 1, pp. 418–425. ACM (2002)
50. Nijhout, H.F.: A comprehensive model for colour pattern formation in butterflies. Proc. Royal Soc. Lond. B Biol. Sci. **239**, 81–113 (1990)
51. Nilsson, D.E., Pelger, S.: A pessimistic estimate of the time required for an eye to evolve. Proc. Royal Soc. Lond. Ser. B Biol. Sci. **256**(1345), 53–58 (1994)
52. Prusinkiewicz, P., Lindenmayer, A.: The Algorithmic Beauty of Plants. Springer, New York (1990)
53. Salazar-Ciudad, I., Garcia-Fernández, J., Solé, R.: Gene networks capable of pattern formation: from induction to reaction-diffusion. J. Theor. Biol. **205**(4), 587–603 (2000)
54. Salazar-Ciudad, I., Jernvall, J.: A gene network model accounting for development and evolution of mammalian teeth. Proc. Natl. Acad. Sci. U. S. A. **99**, 8116–8120 (2002)
55. Sayama, H.: Decentralized control and interactive design methods for large-scale heterogeneous self-organizing swarms. In: Proceedings of the 9th European Conference on Advances in Artificial Life, pp. 675–684. Springer (2007)
56. Sayama, H.: Swarm chemistry. Artif. Life **15**(1), 105–114 (2009)
57. Sayama, H.: Seeking open-ended evolution in swarm chemistry. In: Artificial Life (ALIFE), 2011 IEEE Symposium on, pp. 186–193. IEEE (2011)
58. Schlosser, G., Wagner, G.P.: Modularity in Development and Evolution. University of Chicago Press, Chicago (2004)
59. Schramm, L., Jin, Y., Sendhoff, B.: Emerged coupling of motor control and morphological development in evolution of multi-cellular animats. In: Advances in Artificial Life. Darwin Meets von Neumann, pp. 27–34, 2011
60. Shapiro, B.E., Levchenko, A., Meyerowitz, E.M., Wold, B.J., Mjolsness, E.D.: Cellerator: extending a computer algebra system to include biochemical arrows for signal transduction simulations. Bioinformatics **19**(5), 677–678 (2003)
61. Siero, P.L.J., Rozenberg, G., Lindenmayer, A.: Cell division patterns: syntactical description and implementation. Comput. Graph. Image Process. **18**(4), 329–346 (1982)
62. Stanley, K.O., Miikkulainen, R.: A taxonomy for artificial embryogeny. Artif. Life **9**(2), 93–130 (2003)

63. Turing, A.M.: The chemical basis of morphogenesis. Philos. Trans. Royal Soc. Lond. Ser. B Biol. Sci. **237**, 37–72 (1952)
64. Ulieru, M., Doursat, R.: Emergent engineering: a radical paradigm shift. Int. J. Auton. Adapt. Commun. Syst. **4**(1), 39–60 (2011)
65. Ulieru, M., Unland, R.: Emergent e-logistics infrastructure for timely emergency response management. In: G. Di Marzo Serugendo et al. (eds.) Engineering Self-Organising Systems: Nature Inspired Approaches to Software Engineering, pp. 139–156. Springer, Berlin (2004)
66. Vicsek, T., Czirók, A., Ben-Jacob, E., Cohen, I., Shochet, O.: Novel type of phase transition in a system of self-driven particles. Phys. Rev. Lett. **75**(6), 1226–1229 (1995)
67. Watson, R.A., Pollack, J.B.: Modular interdependency in complex dynamical systems. Artif. Life **11**(4), 445–457 (2005)
68. Webster, G., Goodwin, B.C.: Form and Transformation: Generative and Relational Principles in Biology. Cambridge University Press, Cambridge (1996)
69. Whitesides, G.M., Grzybowski, B.: Self-assembly at all scales. Science **295**, 2418–2421 (2002)
70. Winfield, A., Harper, C., Nembrini, J.: Towards dependable swarms and a new discipline of swarm engineering. In: Swarm Robotics, pp. 126–142, 2005
71. Wolpert, L.: Positional information and the spatial pattern of cellular differentiation. J. Theor. Biol. **25**(1), 1–47 (1969)
72. Wolpert, L., Beddington, R., Jessell, T., Lawrence, P., Meyerowitz, E., Smith, J.: Principles of Development, vol. 3. Oxford University Press, Oxford (2002)
73. Young, D.A.: A local activator-inhibitor model of vertebrate skin patterns. Math. Biosci. **72**(1), 51–58 (1984)