

Robustness and the Halting Problem for Multi-Cellular Artificial Ontogeny

Alexandre Devert, Nicolas Bredeche, Marc Schoenauer

▶ **To cite this version:**

Alexandre Devert, Nicolas Bredeche, Marc Schoenauer. Robustness and the Halting Problem for Multi-Cellular Artificial Ontogeny. IEEE Transactions on Evolutionary Computation, 2011. inria-00566879

HAL Id: inria-00566879
https://inria.hal.science/inria-00566879

Submitted on 17 Feb 2011

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Robustness and the Halting Problem for Multi-Cellular Artificial Ontogeny

Alexandre Devert, Nicolas Bredeche, Marc Schoenauer

***Abstract*—Most works in Multi-Cellular Artificial Ontogeny solve the halting problem by arbitrarily limiting the number of iterations of the developmental process. Hence, the trajectory of the developing organism in the phenotypic space is only required to come close to an accurate solution during a very short developmental period. Because of the well-known opportunism of evolution, there is indeed no reason for the organism to remain close to a good solution in other situations: if the development is continued after the limiting bound; if the environment is perturbed by some noise during the development; if the development takes place in different physical conditions. In order to increase the robustness of the solution against such hazards, a new stopping criterion for the developmental process is proposed, based on the stability of some internal energy of the organism during its development. Such adaptive stopping criterion biases evolution toward solutions in which robustness is an intrinsic property. Experimental results on different "French flag" problems demonstrate that enforcing stable developmental process makes it possible to produce solutions that not only accurately approximate the target shape, but also demonstrate near-perfect self-healing properties, as well as excellent generalization capabilities.**

***Index Terms*—Multi-cellular artificial ontogeny, robustness, generalization, self-healing.**

I. Introduction

Evolutionary Design (ED) is concerned with the application of evolutionary paradigms to the automatic synthesis of complex structures, such as real-world static objects, robot morphologies, or graph topologies. The motivation behind Evolutionary Design is to offer to the human designer a much larger design space than more traditional methods, and allow her/him to explore unexpected design ideas while only specifying a quality criterion, aka fitness function. Indeed, Evolutionary Algorithms are well-known for their flexibility, and their ability to explore huge and poorly-structured search spaces.

Perhaps the most well-known illustrative example of the capabilities of ED is Karl Sims' seminal work [74], in which he evolved creatures made of connected blocks and active joints. Both familiar and original designs were discovered, optimizing ground locomotion, swimming, and target following. While a human engineer may eventually come up with relevant morphologies, inspired from nature or imagination, it is quite unlikely that he/she would provide such a diversity of both original and efficient designs. Another impressive result of ED is that of NASA satellite antennae [56], that actually ended up being more efficient and more compact than human designed alternatives, and were integrated in an actual satellite and sent to space. Other famous real-world achievements range from automatic design of real world crawling robots for locomotion [54], [71] to electronic circuits [31], [82], to cite a few. In all those works, the designer only provided the fitness of candidate designs . . . and the representation, i.e., the specification of search space.

The recent trend in ED regarding representations has been a shift towards developmental approaches, often referred to as Artificial Ontogeny (AO) [8], [79]. Rather than evolving the solution, AO optimizes a program (or developmental process), that will in turn build the solution. The main motivation for such shift is to address the scalability issue (see discussion in Section II). But as inspiration from biology becomes deeper and deeper (e.g., involving complex phenomena like regulation, differentiation, reaction-diffusion of proteins, etc), developmental approaches will involve more and more complex simulations, and will hence be more and more error-prone.

Interestingly, this shift of ED toward developmental approaches meets some concerns of Design in general regarding the buildability of proposed designs. Indeed, the primary desired feature for an automatized design process is its ability to provide efficient blueprints. However, the ability to describe the actual building process is also an important feature. Regarding blueprints, the main issues are expressivity and scalability, so as to possibly provide complex large-scale designs. As for the building process, it should describe *how* to build an object in an efficient way to guarantee that the actual construction is possible in any circumstance (e.g. by avoiding deadlocks). From this viewpoint, a key issue is then related to robustness, i.e., the ability for the design process to be flexible with regards to small mistakes, or missing elements. Such feature is especially important when there are strong interactions between the design process and the environment (e.g., designing a bridge must take into account both general mechanical considerations and topological properties of the environment at hand).

Hence both from the Design perspective and from the Computer Science perspective, the robustness of the building/development phase is an important aspect of the design process. It is however rarely taken into account in the way ED algorithms are developed, being in general only a posteriori studied. Along these lines, this paper presents an indirect way to enforce the robustness of multi-cellular developmental systems by introducing an adaptive stopping criterion for the

A. Devert is with the NICAL, University of Science and Technology of China (USTC); e-mail: marmakoide@yahoo.fr

N. Bredeche is with University Paris-Sud and M. Schoenauer is with INRIA Saclay–Ile-de-France. Both work at TAO, LRI (UMR CNRS 8623), Univ. Paris-Sud, F-91405 Orsay, France - e-mail: Nicolas.Bredeche@lri.fr – Marc.Schoenauer@inria.fr

The authors wish to heartily thank the anonymous reviewers and the non-anonymous guest editor of this Special Issue for the time they spent on the paper, and their valuable comments.

developmental process. The impact of this criterion on the robustness of the underlying ED algorithm is experimentally studied on some "French flag" problems [84], [62], focusing on the sensitivity to perturbation of the development itself (i.e., self-healing in the presence of noise) and scalability with respect to the environment. The main goal of the paper is to explore the link between the halting problem and the robustness of the developmental sequence, and to assess the usefulness of a stopping criterion enforcing a stable development sequence by demonstrating that it ensures robustness while preserving the solution quality.

The paper starts with a general overview of ED challenges and issues in Section II, then focuses on the multi-cellular approaches, and details the flag problems in Section III. Section IV introduces the energy-based stopping criterion. Experiments on different instances of the flag problem are provided in Section V, where the cell controllers are evolved by the standard *NEAT* algorithm, and in Section VI, using an original diffusion-based control. Section VII provides some comparative results involving other stopping criteria from the literature, and the paper ends with some conclusions and hints for further research.

II. Towards developmental encodings

The use of evolution in design dates more than 40 years ago, as summarized in John Frazer's impressive book [26]. However, the field of Evolutionary Design has really started to draw interest from the EC community in the middle of the 90's. In all ED approaches, the goal is to find some objects, structures, or shapes that fulfill some physical requirements. The *phenotypes* of the evolutionary process are hence the descriptions of those structures. But ED approaches can be distinguished by the type of *genotypes* they evolve, i.e., the type of encoding that is used to represent the phenotypes.

Direct encoding refers to encodings where the genotypes are very close to the actual descriptions of the structures. The representation is thus quite explicit, and, more importantly, is human-understandable. There is a strict matching between the elements of the actual construction and the elements of the genotype. However, when using direct encoding, the morphology of a four-legged robot would be encoded with four repeated definitions of a leg, and one definition of a body.

Indirect encoding considers more compact representations, and the genotypes needs to be somehow interpreted to generate an actual construction plans. Though the genotypes are in general not human-readable, indirect encodings generally allow some re-use of elements. For instance, the leg of a four-legged robot could be encoded only once, and re-used four times. Of course, using a more compact description will lead to a high number of inter-dependencies between the elements of the encoding, and will thus tend to violate the *strong causality* principle, i.e., increase the probability that one 'small' genotypic variation can lead to a very 'large' modification of the whole structure. However, it is hoped that the benefits in term of scalability will dominate the drawbacks.

While direct encodings provided an efficient way towards building real world objects due to their intrinsic simplicity, indirect encodings looked more promising with regards to scalability. At the end of the 90's, both direct and indirect encodings were explored at the same time. However, because direct encoding are close from the human engineer's view, they were used by most of the early achievements in ED, including two- and three-dimensional objects such as Lego bridges and cantilever structure [27], [28], [67], tables, cars, optical prisms and various 2D or 3D assemblies of cubic elements [5], tensegrity structures [66], wind turbine blades [18], [70], as well as the examples already described in the Introduction, that were actually built: antennae [56], and the Golem crawling robot [71], [54].

However, such direct representations suffer from several drawbacks. Firstly, the size of the genotype is the number of elements of the phenotype. Indeed, most of the works cited above feature only small assemblies - the Golem crawling robots [54], for instance, are composed of only a dozen of elements. Some more compact representations involving higher levels of descriptions have been proposed (e.g., Voronoi diagrams for chairs [35], [36], problem-specific rotation-oriented representation for optical fibers [57]). However, such representations, that can be seen as intermediate between direct and indirect representations, did not address [35], [36], or only partially addressed, in an ad hoc fixed way [57]) the second limitation of direct representations: the lack of modularity and re-use of elements. In the meantime, several authors [4], [39], [53] had identified and discussed the need for more powerful representations, and advocated indirect encodings. More precisely, several key features were identified that would guarantee efficient encodings (as stated in [39], [53]):

Modularity: well-identified localization of a specific (functional or structural) element;

Regularity: repetitions, or at least similarities, observed in the description;

Hierarchy: recursive composition of a structure and/or function.

Developmental representations are an answer to those arguments: the basic idea is to evolve a construction process rather than a construction plan, putting the emphasis on dynamic building systems rather than static representations. A first consequence is that the genotype length is now related to the structural and/or functional complexity of the target phenotype, rather than to its size.

The definition of this approach, often referred to as Artificial Embryogeny [4] or Artificial Ontogeny [8], [79] ("AO" for short), is two-fold depending on the nature of the building process encoded within the genome (definitions are inspired from [4]):

Explicit Artificial Ontogenies directly evolve the construction process as a program. Early famous examples include Cellular Encoding to grow Artificial Neural Networks [33], and Genetic Programming for the design of analog circuits [47]: the genotype is a list of instructions that are interpreted to grow direct acyclic graphs. Cellular Encoding has been mostly used to grow Artificial Neural Network for various robotic control tasks such as hexapod walk [32], [45] or inverted double pole balancing [34].

Implicit Artificial Ontogenies evolve a set of simple

rules which are then iteratively applied to each element of the growing solution. As opposed to explicit ontogeny, the growing process is here *distributed* over the elements, something that might greatly impact both performance and scalability [4]. The two main alternatives to implicit AO to-date use either grammar-based encodings or more biologically-inspired "cellular" representations:

- **Grammar-based** or **L-system based** Developmental Design use Context-Free-Grammar, of L-systems (formal systems that share many similarities with formal grammars [52]) and provide an easy way to recursively build complex structures. While originally used for simulating artificial plant development, evolvable L-systems or, more generally, grammar-based developmental systems, have been applied in the seminal work of Karl Sims to evolve the morphologies of artificial creatures [74], work that had some recent followers [59], [51], [48]. Other application regard Lego structures [68], [69], table design [38], robot design [40], antennae design [55] (as a follow-up to [56]) and architecture form generation by combining a variation of L-system with Grammatical Evolution [65].
- **Multi-Cellular** Developmental Design takes an even tighter inspiration from developmental biology, mimicking the way cells grow, migrate, divide, differentiate, and regulate and be regulated by other cells from the organisms. Recent achievements include variations over Genetic Regulatory Networks to design 3D lens shape [21] and box-pushing robot [8]; optimization of various kind of Cellular Automata for designing skyscrapers [42], [43], [44]. Also, recent works investigated artificial ontogenic processes in interaction with the environment such as growing and/or stabilizing metallic truss structures under mechanical stress [46], [16], [81] and environment-guided development of heat-protecting walls [22].

Some works cited above are explicitly based on a distributed variant of Gene Regulatory Networks (GRN) for automatic conception of robot morphologies [19], [8]. However, it should be noted that most works within the Evolutionary Computation that use the GRN paradigm (e.g. [72], [7], [20], [3], [49], [50], [17], [73], [58], [63]) do not pertain to multi-cellular developmental design, as the resulting network is created in a centralized fashion through a one-pass interpretation process of the artificial genome, in contrast with the distributed construction within the multi-cellular developmental approaches.

Of course, this taxonomy bears some limitations, and does not include all works that reclaim from Artificial Ontogeny, such as algorithm that actually embed an ontogenic process based on heuristics during the genotype-to-phenotype mapping (e.g. neural connection growth depending on external stimulation [64]). A noteworthy exception is the HyperNEAT algorithm [77]. This algorithm has already provided impressive results and interestingly stimulating discussion as to what level of abstraction is relevant when considering Artificial Ontogeny. HyperNEAT successfully captures key features from Artificial Ontogeny, but in a static way: NEAT-optimized Compositional Pattern Network (CPPN) [76] “develop” in a one-pass genotype to phenotype mapping, bypassing the temporal developmental sequence featured in Artificial Ontogeny works. While this may be limited whenever strong interactions with the environment are required, the issue of the relevance of the price of the temporal developmental sequence remains open, and may well depend on the problem at hand.

III. Robustness and the Flag problem

A. Issues in Developmental systems

As stated in previous Section, a developmental system evolves a construction process, rather than a plan. In other words, the genotype also encodes the mapping process from genotype to phenotype. Furthermore, this mapping can be viewed as a dynamical system in the space of phenotypes, and the goal of evolution is then to tune this dynamical system toward an end point (a phenotype) that is somehow optimal for the fitness at hand. A first issue is that of the Halting Problem for this dynamical system. But other important issues from the applicative point of view are the various robustness issues, with respect to the initial/varying environmental conditions (i.e., what is the influence of error/noise during development?), and with respect to scalability (i.e., what would be the result of applying the same ontogeny process with larger environmental resources like energy, size, etc?). Of course, if neither scalability nor noise- and fault-tolerance are issues, a developmental encoding can be simply viewed as some kind of compression process of the phenotypes. Otherwise, addressing robustness issues requires some stability and generalization properties of the dynamical system with respect to environmental and experimental conditions: to some extent, the result of the developmental process should not vary abruptly under slightly modified conditions. Roughly, this distinction can be formulated as comparing careful design of precise trajectories (i.e., uncompressing process) and designing basin of attractions towards approximate or exact solutions (i.e., developmental solutions with generalization abilities).

The scope of this work is to study the robustness issue of Multi-Cellular systems within the particular framework of the so-called “French Flag” problem. This problem takes inspiration from developmental biology with the study of the influence of morphogens in the development of a cell [85], [86]. This setup has been used over the years in various contexts in Computer Science [52], [2], [37], and has been introduced to Evolutionary Design by Miller [60] as a privileged experimental sandbox for Multi-Cellular Developmental Systems. It provides a clear and well identified setup which makes it possible to study specific issues of general interest to Developmental Design.

B. The Multi-Cellular Flag Problem

The Flag problem can be defined as an optimization problem where the goal is to find the update rules for a two-dimensional continuous Cellular Automata so that the whole CA state converges to a target pattern (e.g. French, Norwegian, Japanese flags, geometric shapes, etc). Given the loose inspiration from real world embryogeny, the system is usually described using a biologically-inspired lexicon: an *organism* is made of identical units, aka cells, that are spatially arranged with a

TABLE I
PROPERTIES OF REPRESENTATIVE WORKS WITH THE MULTI-CELLULAR FLAG PROBLEM (NON-EXHAUSTIVE LIST).

	Miller et al. [60]	Gordon et al. [29]	Federici et al. [23]	Chavoya et al. [11]
setup	32×32, 9 steps	20×20, n/a	32×32, 12 steps	33×33, 100 steps
states	binary string	binary string	real valued vector	1 bit and 2 reals
neighbourhood	8	4	8	8
update function representation	Boolean logic circuit and diffusion	implicit boolean transition table	Multi Layer Perceptron and diffusion	Gene Regulatory Network
cell states update order	asynchronous, deterministic order	N/A	asynchronous, cell age order	asynchronous, random order
ontogeny control	fixed number of development steps given by user	stop when steady-state reached and fixed number of development steps given by user	development carried through stages, fixed number of development steps given by user	self-tuned (evolved) or fixed number of development

given topology (only fixed neighborhoods will be considered here). Cells communicate with their neighbors, exchanging vectors of real values, termed *chemicals*: A given cell sends chemicals to nearby cells and, in turn, receives chemical vectors from each of its neighbors. Global information is thus only transmitted through local interactions. In particular, cells are not aware of their position, though some cells may be used to bootstrap global positioning because of specific positions (e.g. cells at the borders or at the corners of the environment). To some extent, this setup can be related to the reaction-diffusion models [83], with the notable difference that cells are physically modeled (i.e., the environment is a discrete 2D grid rather than a continuous substrate). The motivation behind using the flag problem is that it provides a common ground for studying crucial features on artificial developmental processes, while remaining simple enough for visualization and understanding purposes, and yet still being able to yield complex dynamics.

Figure 1 gives an example of environment and cell input/output, assuming the topology is defined with von Neumann neighborhoods. Time is discrete, and the updates of the cells are synchronous. Of course, there exist many other ways to define the environment, with regards to the neighborhood (e.g. von Neumann, Moore, Margolus, totalistic or not, etc), the update scheme (synchronous, asynchronous, etc.) and the developmental flavor (with or without spatial development). The development of an organism is illustrated in figure 2: the whole organism starts from an initial state where all cells are undifferentiated, and ends to a shape that is close to the half-discs target (see Section V-B). In this example, differentiation starts from border cells and diffuses to the center of the organism[1]. Then again, various approaches have been explored regarding the update rule, including Cartesian GP [60], Artificial Neural Networks [23], [14], [15], rule-based [29], [10] and GRN-inspired [9], [6], [11], [25]. Table I gives a summary of the main properties of major contributions to the multi-cellular flag problem.

In this work, as in many others, the fitness (to be minimized) is the distance between the fully developed organism and the target image, formally defined as:

$$\sqrt{\frac{\sum_{x=1}^{\text{width}} \sum_{y=1}^{\text{height}} (I_{\text{org}}(x,y) - I_{\text{target}}(x,y))^2}{\text{width*height}}} \quad (1)$$

where I_{org} is the image described by the organism once development has come to a halt, and I_{target} the target image, both of them being width*height matrices of gray levels in $[0,1]$ (the fitness hence also lies in $[0,1]$).

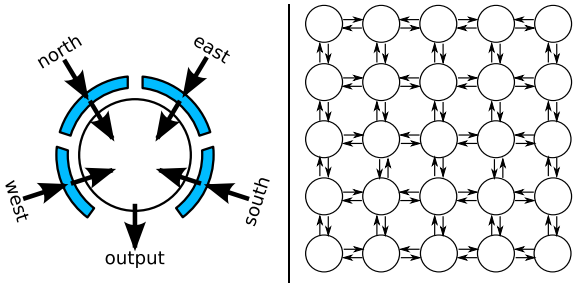


Fig. 1. Cell input/output (left) and von Neumann neighborhood topology within a population of homogeneous cells in a $2D$ environment (right).



Fig. 2. A developmental sequence (target: "half-discs")

C. Halting the developmental process vs. Robustness

As already mentioned, time is considered discrete: during one developmental step, all cells update the quantities of chemical they will send to their neighbors at next step, based on their current state and the chemicals received from their neighbors. A critical issue is hence when to stop such developmental process.

One of the simplest stopping criteria for the development phase is to a priori set a number of development steps, as was done in most of the previously cited works (see Table I). However, one of the motivations for using Artificial Ontogeny approaches is their promise for robust and scalable solutions (Section III-A above). In the context of the Multi-Cellular developmental systems applied to the Flag problems, scalability can be seen as the ability of the system to reach

[1]It should be noted that development is defined as a temporal process of differentiation through time, and does not automatically imply (while it may also be possible) a spatial process of organism growth through cell duplication.

the same target pattern at several scales. An essential step toward scalability is achieved by using the same controller and the same initial state for all cells: the genotype size is then independent from the pattern size. But larger patterns also imply longer development times: it is very unlikely that the same number of development steps will be required for development at different scales (e.g. from 16×16 to 32×32 environments) and a fixed number of development steps may thus hinder the scalability properties of the method.

Some works have explored alternative stopping criteria, such as self-tuning of the number of developmental iterations [10] (i.e., include it into the genotype and let evolution tune it), or stopping whenever a fixed point or limit cycle is detected before some maximum number of iterations [30] (but relying on randomness, i.e., without explicitly enforcing the occurrence of such a stable state). It should also be noted that works in different field of Artificial Ontogeny (e.g., modeling artificial Gene Regulatory Networks) also tackled the issue of evaluating development stability during a limited period of time [62], [80]. These approaches relied on minimizing target matching during *several* arbitrary chosen iterations to favor transient (or possibly definitive) stable configurations (see Section VII for further discussions).

But another important issue is that of robustness with respect to perturbations of the initial/developmental conditions. When considering a fixed number of developmental steps, such robustness can only be assessed a posteriori [61], [24]: Some of the solutions obtained in those works do admit a steady-state of the developmental process, acting as a loose attractor: carrying on the development after random perturbations of the organism does produce patterns resembling the initial target (e.g. the 3-bands French flag), though in some kind of degenerated ways (repeated encapsulated patterns, or stretched patterns). However, while these experiments gave promising results with respect to self-repairing, they failed to demonstrate full recovery within different initial and/or experimental conditions. Our claim is that this is because the stability of the dynamical developmental process is not enforced during evolution, and the proposed approach aims at incorporating stability in the developmental process through its stopping criterion.

IV. From Evolved Stability to Robustness

When considering the developmental process of a given multi-cellular organism as a dynamical system in the space of all possible phenotypes, the above limitations of setting a fixed number of developments steps as stopping criterion can be rephrased as follows. The only requirement on the dynamical system is that the trajectory of the developmental process is as close as possible from the target phenotype at a given point in time, without any constraint on the complete trajectory. In particular, there is no reason for the system to remain close to the target pattern after the required number of development iterations. Furthermore, there is no reason for the trajectory to be stable, in the sense that the system would stay close to the original trajectory after some random perturbation during the development process, or come back to the target phenotype if the phenotype is perturbed after development is complete. And of course, as already argued, there is little chance that the trajectory comes close to the target phenotype when the physical environment is changed, e.g. grid scale is modified, as the number of iterations required to reach a given pattern obviously depends on the size of the pattern pixel-wise. To summarize, in multi-cellular approaches that use a fixed number of iteration steps for the developmental process, there is no incentive whatsoever for robustness, be it robustness against noise during the development process, robustness against perturbations posterior to the development (aka self-healing), or robustness with respect to the environment. And because evolution is known to be opportunistic, if there are ways to minimize the fitness that are more easily reachable in the genotypic space than the ones that are robust, evolution will favor these.

Because the robustness of the developmental process is clearly linked to the stability of the underlying dynamical system in the phenotypic space, one way to reach stability is to favor, through the selection process, those dynamical systems that converge to a fixed point (i.e., a stable configuration), or at least that reach a stable basin of attraction. Robustness issues can then be reformulated as sensitivity to initial conditions and perturbations (stability then depends on the width of the basin of attraction), and dependency on the experimental conditions (convergence towards a stable state should hold if the scale is changed – though the resulting fixed point of convergence might slightly vary).

A. Energy-based stopping criterion

The mechanism that is proposed here in order to reach the dynamical stability was first described in [14]: during the course of evolution, only genomes leading to stable configurations are considered. Stability is determined according to the cell activity: an *"energy level"* is computed at each time step, measuring the variations of input/output chemical concentrations over a time window. The activity of the organism is defined as the total variation of the energy levels of all cells over time. It is expected that the organism has reached a stable state (or a stable basin of attraction) whenever the activity is stable (or varies very slowly).

Along this line, an *energy measure* of the multi-cellular organism is computed at each developmental step. If the variations of the energy value of the organism are very small over some time window, the development is stopped, as it is likely that development reached a steady state. In the current setup, a simple and a general formulation of energy can be devised from the sum of the squared values of all the cell states. A steady state is detected from the standard deviation of the energy over a time window (up to some tolerance): if the standard deviation is lower than a given threshold ε, development is considered to have reached a steady state and halts.

More formally, the proposed stopping criterion is formulated as follows. Let $state(c,t)$ be the state of cell c at time t, defined as the vector of both internal (e.g. neuron activities here, see Section V) and external (e.g. input and output

chemical concentrations) values. The energy of the organism, made of N_{cells} cells, at a given time t, is given by:

$$e(t) = \sqrt{\sum_{c=1}^{N_{cells}} ||state(c,t)||^2} \quad (2)$$

where $||.||^2$ the Euclidean norm in the state space.

The average $\overline{e}(t)$ and standard deviation $\mathcal{V}ar(t)$ of the energy of the organism at time t are then computed using a time window of size k:

$$\overline{e}(t) = \frac{1}{k} \sum_{i=t-k}^{t} e(i) \quad (3)$$

$$\mathcal{V}ar(t) = \sqrt{\frac{1}{k} \sum_{i=t-k}^{t} \left(e(i) - \overline{e}(t)\right)^2} \quad (4)$$

The development then stops at time t whenever $\mathcal{V}\text{ar}(t) < \varepsilon$, for a user-defined threshold ε.

In practice, a time window of size at least $k = 16$, with a tolerance $\varepsilon = 10^{-15}$, were experimentally found sufficient for all experiments presented in the following. Furthermore, as expected, all organisms that were evolved using this stopping criterion actually remain stable even if many more iterations were allowed.

After the energy measure has been defined, the issue of enforcing stable development within a finite time is yet to be addressed. Indeed, some candidate controllers might lead to periodic or chaotic dynamics, and no steady state will ever be reached in such a case (this is particularly true in the early generations). In order to address this issue, an upper bound for the maximum number of development steps is enforced. This bound is one order of magnitude above the number of iterations that would be necessary for all cells to communicate. For instance, for the 32×32 patterns considered in the following, the upper bound is set to $T_{max} = 1024$, i.e., long enough to allow a signal to travel 32 times from one side to the other of the organism. Organisms for which the developmental process never triggers the stopping criterion within the maximum allowed number of development iterations are heavily penalized, and are likely be eliminated in next selection steps, at least if some other genotype do fulfill the developmental stopping criterion.

Note that the stability is only indirectly enforced in the fitness, as no explicit (un)stability measure is added to the fitness, introducing an additional parameter to balance between matching accuracy and stability. Here, the accuracy fitness is only computed after stability has been reached: The underlying hypothesis is that stable developmental sequences must be enforced even before the matching error is minimized. Evolutionary search for accuracy only occurs within stable phenotype, in order to address robustness issues.

B. Methodological and Experimental Roadmap

The rest of the paper aims at experimentally validating the energy-based stopping criterion. Next two Sections describe a set of experiments based on this stopping criterion using two controller optimization methods. The primary motivation is to evaluate the relevance of the stopping criterion with regards to robustness to noise, i.e., checking the variations of the results when adding noise in the experimental conditions of the development, and scalability, i.e., running the development in larger environments without any further optimization.

The developmental model used hereafter is a cellular automaton on a fixed 32×32 grid, with von Neumann neighborhood. All cells start with the same initial condition (there is no *spatial* growth). Each cell is endowed with an update rule modeled after a discrete-time (possibly recurrent) Artificial Neural Network (ANNs) with sigmoidal activation function that takes as input the concentrations of chemicals from its four neighboring cells, and provides continuous output values for chemical concentrations and differentiation (gray scale level determining the final 'color' of the cell for comparison with the target 'flag'). The genotype of the organism describes the neural network, that is common to all cells. All cells also share the same initial activation: the model is that of an homogeneous continuous automaton. The choice of ANNs as controllers is motivated by their well known theoretical properties [41] and by the availability of efficient and widely recognized neuro-evolution algorithms [78].

Two neuro-evolutionary algorithms will be under scrutiny. First, the state-of-the-art NEAT algorithm (Section V), which is recognized as a standard for comparison; Second, a variation of a simple neural-network based approach where the focus is put on heuristics to build up relevant inputs, inspired from reaction-diffusion systems (Section VI). In each of these approaches, robustness issues are investigated and discussed. In both setups, the evolved part is the cell controller itself, i.e. the weights and/or topology of the neural network that maps cell inputs (i.e. incoming chemical concentrations) to cell outputs (i.e. emitted chemical concentrations and color).

Investigating two different controller types is motivated by several considerations. Firstly, the relevance of the energy-based stopping criterion with respect to efficiency and robustness must be clearly identified and separated from possible bias from the controller model. Secondly, the intrinsic mechanisms required, and possibly evolved with NEAT are investigated from a different perspective by considering reaction-diffusion communication scheme and a simple multilayer Perceptron as controller. This enables to pinpoint the main difficulty when optimizing the cell update function (i.e., the communication scheme) and also provides some insights into the balance in the trade-off between what can be hand-coded in the model and what should be left for evolution to optimize.

Next two Sections V and VI follow the same overall organization, starting with a description of the model and the optimization method, then presenting the particular elements of the experimental setup, and finally discussing the results. Lastly, Section VII provides an in-depth study of the energy stopping criterion, focusing on the specific dynamics of the developmental systems. In particular, development with a fixed number of iterations, a usual practice in the literature, is revisited from this perspective. Other alternative methods

are reviewed, too, and elements of comparison are provided. Importantly, the hypothesis underlying the present work, that relates stable development to robustness, is validated and discussed.

V. Neural networks as update functions: NEAT

This Section presents a first set of experiments, using *NEAT* algorithm [75] to optimize both the topology and the weights of the discrete-time recurrent ANN that controls all cells in the fixed-layout organism.

A. The model

At a given development step, each cell receives some inputs (chemical concentrations from neighbor cells), updates its internal states (the activation values of all neurons of its own ANN controller) and emits outputs (the chemical concentrations it will send to its neighbors at next step, and its 'color'). The chemical concentrations are received and emitted instantaneously (there is no delay between steps), and the cell 'color' is a single real value that is interpreted as its color, to be compared to the corresponding pixel color of the target shape (Equ. 1). All values lie in $[-1, +1]$, except for the cell color, which is linearly normalized in $[0, 1]$. Initially, all chemical concentrations are set to zero. The organism is updated in a synchronous fashion: all cells are updated synchronously, by taking as inputs the current chemical levels of their four neighbors, and using their ANN controller to set the chemical levels they will emit at next step, and update their color. The development stops according to the criterion defined in Section IV-A, i.e., it stops whenever the standard deviation of the energy over the last 16 steps is below 10^{-15}, or the maximum number of steps is reached. The fitness is computed according to Equ. 1, and penalized by a large value if the energy criterion did not trigger before the maximum number of steps.

The optimization of the ANN controller is handled by an Evolutionary Algorithm. However, several practical studies [12], [33], [75] have demonstrated that the topology can have a critical impact on the performance of an ANN, despite the theoretical properties of simple feed-forward Perceptrons. Furthermore, the state-of-the-art Evolutionary Algorithm *NEAT* [75] is easily available, and is well known to provide efficient results on a wide range of tasks. It was hence chosen here.

NEAT enables the optimization of both feed-forward and recurrent topologies. It relies on a direct encoding of neural network topologies, that are evolved using a non-standard evolutionary optimization engine. The main feature of *NEAT* is that it explores the topologies from the bottom-up: starting from the simplest possible topology for the problem at hand, it performs variations by adding neurons and connections to the existing network in such a way that the behavior of the network is preserved as much as possible, thus exploring the space of topologies in a conservative manner.

B. Experimental Setup

In order to explore different flavors of the developmental model at hand, four instances of this model are considered, and optimized using *NEAT*. They differ by the type of ANN (feedforward or recurrent), and the number of chemicals exchanged between cells (1 or 2), and are respectively termed

- *1-ffwd*, for feedforward ANN, using 1 chemical;
- *1-recurr*, for recurrent ANN, using 1 chemical;
- *2-ffwd*, for feedforward ANN, using 2 chemical;
- *2-recurr*, for recurrent ANN, using 2 chemical.

Classically, like for most computational approaches to complex system modeling, there are (at least) two possible causes of error in the proposed approach: the *modeling error* (there might not exist any fixed point of the multi-cellular developmental systems under study that can approximate the target image); and the *method error* (even if a good solution does exist, the evolutionary optimization of a neural network using *NEAT* might not be able to approximate it). *Numerical errors* will be neglected here.

In order to try to discriminate between the modeling error and the method error, a fifth model is also considered: it involves no chemical, nor any exchange of information between neighboring cells. Instead, all cells receive as inputs their (x, y) coordinates on the grid. While this does not qualify as a spatially distributed artificial ontogenic process (as it is fully informed), it nevertheless provides an upper-bound baseline for evaluating all other experimental setups. In the following, the results of this model will be considered as reference results, as it is not expected that any developmental approach can ever beat a totally informed model if using the same optimization tool. This experiment is termed *neat-non-dev* from now on.

All five models described above are run on the four target patterns showed on Figure 3. These targets are loosely inspired from existing experimental settings, and provide challenges of gradual difficulty, from the easy *two bands* target to the more challenging *half-discs* target. All results presented in

TABLE II
NEAT parameters

Population size	500
Max. number of evaluations	250000
Reproduction ratio per species	0.2
Elite size per species	1
Crossover prob.	0.15
Add-node mutation prob.	0.01
Add-link mutation prob.	0.01
Enable-link mutation prob.	0.045
Disable-link mutation prob.	0.045
Gaussian weights mutation prob.	0.8
Std. dev. for Gaussian weight mutation	0.1
Uniform weights mutation prob.	0.01
Distance parameters for fitness sharing	1.0 – 1.0 – 0.2

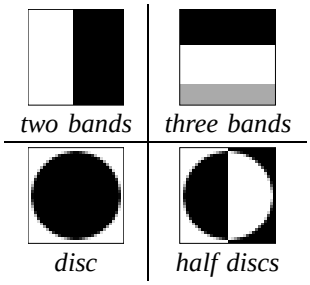


Fig. 3. The four 32×32 targets used for the experiments

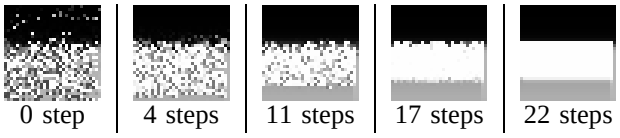


Fig. 4. Self-healing on the three-bands problem for the recurrent NN and 2 chemicals

the following are statistics gathered over 16 independent runs. *NEAT* was used with its default settings, recalled in table II.

It is worth noticing that during all runs, no bloat (i.e. uncontrollable structure growth with no functional advantages) was ever observed within *NEAT* genotypes. The mean size of the networks (measured by the total number of edges between neurons) gently grew from its starting value (between 5 and 10 depending on the model) to some final value below 40 – the largest experiment reaching 45. This first result confirms the robustness of this optimization tool, but also, to some extent, demonstrates the well-posedness of the problems that are given to *NEAT* (bloat during ANN topology optimization can be a sign of over-fitting some ill-described data).

C. Results and discussion

Figure 5 shows that for each target, the method was able to provide stable results. The first consideration is that all runs were successful at finding self-terminating developmental sequences. Also, it could be feared that hundreds of iterations would be needed before stabilization even for the best solutions found by the algorithm due to the hard-coded limitation of 1024 developmental iterations. The fact is that all the evolutionary runs showed that the whole population quickly features a large majority of organisms that do stabilize within a few dozens iterations. For the easy *three bands* problem, around 40 iterations are enough, whereas for the more difficult *half discs* problem, about 120 iterations are needed (and enforced by evolution in all runs).

The coordinate-informed non-developmental approach always provides better matching scores with less evaluations. This was of course to be excepted, since the problem solved through development is by far harder. The non-developmental solution provides an over-confident bound of the difficulty to match a given pattern using a given optimizer.

The *two bands* target seems to be an easy problem: the convergence is very fast for the four settings, and the accuracy quite close to the that of the non developmental approach. Three of the four models reached a pattern matching score of 10^{-3}, which is a qualitatively convincing match.

Results with the *three bands* target are less enthusiastic: the median best solution does not reach the 10^{-3} matching score, although some of the runs actually did. All 4 settings tend to lead to optimization that stalls after $100\ 000$ evaluations around a 2×10^{-2} matching score. In contrast, the non developmental approach can still reach a pattern matching score of 10^{-3} in less than $150\ 000$ evaluations.

Things get even worse for the *disc* target. Even the non-developmental approach does not reach 10^{-3} anymore: the median score is above 10^{-2} after $250\ 000$ evaluations, though the runs are not yet stalled. The developmental approaches reach a score between 10^{-1} and 8×10^{-2}, quite far from the results of the non-developmental approach. Also, convergence is much slower than for the *two bands* and *three bands* targets.

Last, the *half discs* target is by far the most difficult target. The non developmental approach reaches a matching score slightly above 10^{-2}. The developmental approaches do improve a lot the initial candidates solutions, however stalling around 2×10^{-1}.

Solutions have also been evaluated with respect to robustness. At some point during the development, a random perturbation, following a centered Gaussian with unit standard deviation distribution, is multiplicatively applied to the internal states of all neurons, and to the current chemical concentrations. Development then proceeds from the perturbed state. The good news is that for all perturbations, **100%** of the organisms with feedforward controllers and **75%** of the organisms with recurrent controllers converge back to *exactly* the same global state that was reached before perturbation. The remaining 25% of organisms with recurrent controllers do not actually fail to be robust though: they all converge to a global stable state that is rather close to the target state. An example of perfect and fast self-healing for the *three-bands* problem is shown in figure 4.

This Section has demonstrated that it is possible to get reliable results for the developmental system at hand by using the energy-based stopping criterion, with regards to both pure efficiency and robustness to noisy initial conditions. The NEAT algorithm used to provide the ANN controllers for the cell update function performed well on all the benchmarks. However, several issues remain to be addressed. In particular, the impact of using NEAT algorithm must be studied so as to ensure that stable developmental process are not due to a hidden intrinsic bias in the evolutionary process. This issue is addressed in next Section, where NEAT is replaced by yet another neuro-evolution algorithm. Note that this move is also motivated by one of the only drawbacks of NEAT, namely the difficulty to set up its many free parameters.

VI. A DIFFUSION BASED APPROACH

In the previous Section, relying on NEAT made it possible to consider the problem as a black-box optimization problem (i.e. the algorithm provides a solution without any background knowledge from the expert). While it represents the advantage of simplicity from the expert's viewpoint, it also ignores possibly relevant mechanisms and/or principles that could be beneficial to development. Even worse, common required features in the update function will have to be rebuilt for any new problem. This Section investigates breaking the update function into multiple functional units, so as to relieve the optimization process.

One of the candidate general mechanisms for doing so is *diffusion* [60]: diffusion is relevant whenever (linear or radial) gradients and associated variations are required. To some extent, this is similar to what is observed within reaction-diffusion systems. Indeed, the proposed implementation of the diffusion process actually mimics a discretized reaction-diffusion system.

A. The model

A cell state is a pair of two vectors (u, v), where u is the vector of external chemicals, and has M elements, and v is the vector of internal chemicals, with N elements. The internal chemicals of a cell are used to compute the behavior of this cell from the concentrations of nearby external chemicals – replacing to the use of neural networks as controller in previous Section. Here again, the values of all coordinates of u and v are in $[-1, 1]$, and are initially set to 0, and the update function is applied to all cells in a synchronous fashion. The inputs are the external *and* internal chemical concentrations: the internal state remains local to the cell while the external chemicals are received from its four neighbors (von Neumann neighborhood). The update function outputs the new internal and external chemical values, the latter being sent to all neighbors, as follows:

$$\Delta_{i,j}(t) = f_u(u_{i+1,j}(t), u_{i-1,j}(t), u_{i,j}(t), v_{i,j}(t), u_{i,j-1}(t), u_{i,j+1}(t)) \quad (5)$$

$$u_{i,j}(t+1) = \tanh\left(u_{i,j}(t) + \Delta_{i,j}(t)\right) \quad (6)$$

$$v_{i,j}(t+1) = f_v(u_{i+1,j}(t), u_{i-1,j}(t), u_{i,j}(t), v_{i,j}(t), u_{i,j-1}(t), u_{i,j+1}(t)) \quad (7)$$

Both functions f_u and f_v are encoded as a simple feed-forward Perceptron with one hidden layer, using the hyperbolic tangent as activation function. If the Perceptron has H hidden units (and each neuron one bias input), the update function has $(4M + N + 1) \cdot H + (H + 1) \cdot (M + N)$ parameters. When a neighbor is missing (i.e. cells at the border), the missing inputs are set to 0. Furthermore, before computing the cell update function, a Gaussian blur function with a radius of one is applied to incoming external chemical values from neighbors. This low-pass filter stands for a rather crude diffusion process.

The expression function ϕ computes the cell color by interpreting a pre-defined internal state (defined in $[0, 1]$) as a gray level color information, according to the following equation:

$$\phi(u, v) = \frac{1}{2}\left(1 + \tanh\left(s_u \cdot u + s_v \cdot v + s_{bias}\right)\right) \quad (8)$$

This corresponds to a single layer Perceptron with one bias input, where s_u, s_v and s_{bias} are the synaptic weights. As a consequence, the expression function has $M + N + 1$ parameters.

Such a model can be viewed as a time- and space-discretized reaction-diffusion system. Diffusion is approximated by a Gaussian blur pass, and all chemicals have the same diffusion constant. The update function plays the role of the local chemical reaction function, but also acts as a Laplacian operator. As a consequence, it would be possible to tweak the diffusion process, and to allow anisotropic diffusion.

The stopping criterion for the development phase, as before, is based on an energy measure of the whole multi-cellular organism defined in Section IV-A. For this diffusion-based model, the energy of a single cell (see Equ. 2) can also be written as $||u||^2 + ||v||^2$ (and the energy of the whole organism is the again sum of the energy of all its cells).

In the experiments presented below, a single chemical configuration is experimented: the cell states are made of 1 internal and 2 external chemicals. Note that this is much smaller than the 10 to 40 states (neurons activations) per cell that were evolved by NEAT. The update function is a multi-layer Perceptron with 8 hidden units, i.e. genome consists of 111 real values (107 for the update function and 4 for the expression function). The optimization is performed using the out-of-the-box recent version of CMA-ES algorithm [1], the state-of-the-art algorithm today for stochastic continuous optimization. Beside being efficient, CMA-ES also features very robust default parameters that generally do not require any fine-tuning. For each experiment, 32 independent runs are performed, with a maximum of 10^5 fitness evaluations for each run. Compared to the NEAT-based approach, there are very few specific parameters to tune here: M, N and H which controls the complexity (thus the compactness) of the model.

B. Results and discussion

The median, maximum and minimum of the fitness over the number of evaluations for the 4 target patterns are presented in Figure 5. On the simple *two bands* target, the diffusion-based model does not have the strong bias of the *NEAT*-based approach. This is easily explained: the Perceptron used as an expression function in this model relies on a fixed transfer function, the hyperbolic tangent. This function is a rather gentle steep function, so obtaining sharp gray level transitions requires a lot of fine tuning from the optimizer. On the opposite, *NEAT* can build very steep functions by linking neurons in long chains. Evolving a steepness parameter for the expression function may help to solve this issue, but this remains open at the moment. This bias does not appear on the *three bands* target runs, despite the sharp edges: The third gray band is difficult enough to build from the very poor information brought by the fitness function. However, results of the diffusion model are comparable with those of the NEAT-based approaches. Interestingly, on the *disc* target, the diffusion-based model obtains the best results: indeed, a circle is easy to build using isotropic diffusion. Finally, results are also improved over the previous approach on the *half discs* target.

Overall, except for the easy *two bands* target, the diffusion-based model reaches better pattern matching scores with fewer evaluations than the NEAT-based approach. Furthermore, it should be emphasized that the diffusion-based approach reaches similar scores with only three states per cells compared to the 10 to 40 states per cell that resulted from NEAT optimization. Accuracy-wise, the diffusion-based model appears from comparable to clearly superior to the NEAT-based approach, as show on figure 5.

Furthermore, all solutions found with the diffusion-based model also demonstrate excellent stability properties. Robustness to initial noisy condition was shown to be comparable to previous achievements. Small Gaussian noise (with standard deviation of 10^{-1}) added to cell states during the development

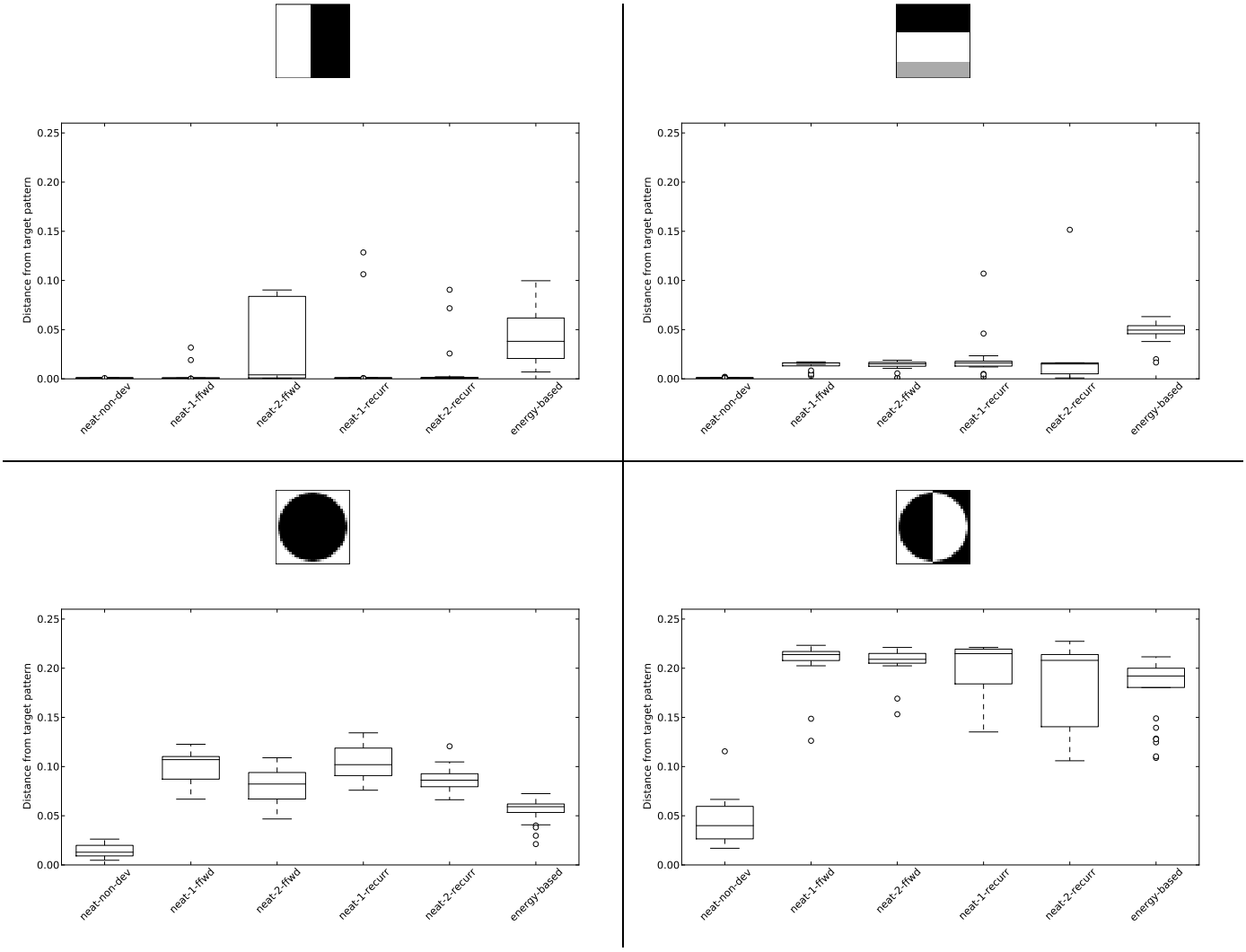


Fig. 5. The boxplots show the distribution of the best pattern matching score obtained by our runs, for each pattern and each approaches. *neat-1-ffwd*, *neat-2-ffwd*, *neat-1-recurr* and *neat-2-recurr* stand for our experiments with NEAT and general neural networks, either feedforward or recurrent, with one or two chemicals. *mlp-diff* stand for the diffusion based approach.

TABLE III
TIME TO STABILIZATION AND SCORES OF THE BEST EMBRYOS OBTAINED WITH THE DIFFUSION-BASED METHOD USING THE ENERGY-BASED STOPPING CRITERION, WHEN THE 4 FIRST STEPS ARE PERTURBED BY AN ADDITIVE GAUSSIAN NOISE OF STANDARD DEVIATION 1.0

	nb steps	similarity
two-bands	444.095 ± 232.781	0.053 ± 0.066
three-bands	343.312 ± 221.968	0.063 ± 0.083
disc	854.656 ± 405.681	0.060 ± 0.012
half-discs	639.0 ± 298.515	0.186 ± 0.037

TABLE IV
TIME TO RE-STABILIZATION AND SIMILARITY SCORES OF THE BEST EMBRYOS OBTAINED WITH THE DIFFUSION-BASED METHOD AND THE ENERGY BASED STOPPING CRITERION, WHEN THE 4 STEPS AFTER STABILIZATION (I.E. STEPS 1024, 1025, 1026 AND 1027) ARE PERTURBED BY AN ADDITIVE GAUSSIAN NOISE OF STANDARD DEVIATION 1.0. IN THIS CASE, THE FULL DEVELOPMENT SEQUENCE IS ALLOWED TO LAST 2048 STEPS (NUMBER OF STEPS TO CONVERGENCE IS COUNTED FROM ITERATION 1028).

	nb steps	similarity
two-bands	442.25 ± 224.797	0.042 ± 0.025
three-bands	315.406 ± 200.433	0.059 ± 0.051
disc	706.094 ± 236.428	0.0560 ± 0.011
half-discs	693.226 ± 308.911	0.183 ± 0.032

only slows down the convergence, but the resulting pattern is nevertheless close from the target pattern. If some larger noise is added (standard deviation of 1.0), the flag quickly recovers the target pattern. Tables III (resp. IV) show the time to convergence and final matching score in the context of perturbations at the beginning (resp. end) development. In both cases, not only all embryos recover from perturbations, but timings and scores are also comparable to what was obtained in the unperturbed conditions. Even though the qualitative and quantitative results remain mostly stable (as illustrated by Figure 6), the dynamics may also be slightly different, and some organisms might converge to different attractors (as illustrated by Figure 7) even though resulting in a pattern very similar to the target. To a lesser extent, this latter phenomenon was also observed with the *NEAT*-based model (results not shown here).

The diffusion-based model features stable development, comparable or even better than with the NEAT approach, but with much fewer parameters and faster convergence – this advocates the relevance of the diffusion process as a

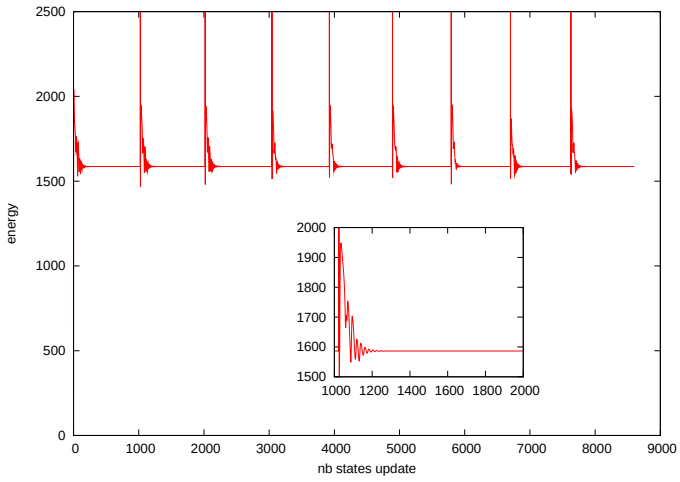


Fig. 6. Diffusion-based method: Energy of the cells of a 32×32 flag, controlled by the best controller found during one of the optimization runs. The target pattern is the *half-discs* pattern. Each time the variance of the energy over 16 steps is bellow 10^{-15}, a zero centered Gaussian noise with deviation 1 is added to all states of all cells, resulting in an energy spike. After such bursts, the energy always returns to the same level, with a damped oscillations dynamic. Thus, the optimized controller seems to have a very strong attractor, encoding the final solution.

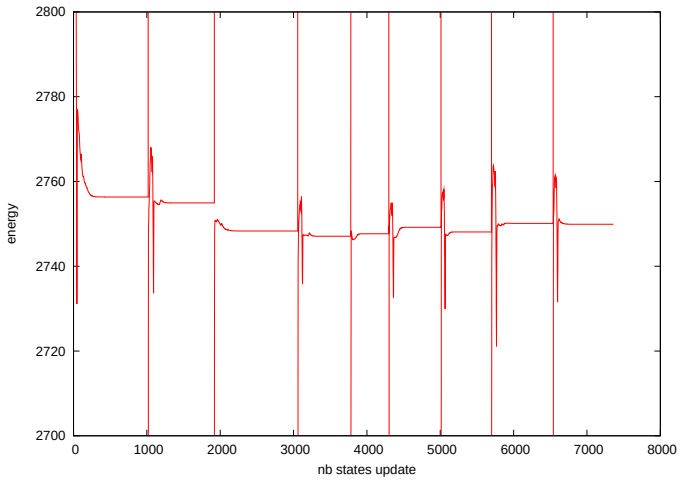


Fig. 7. Diffusion-based method: Energy of the cells of a 32×32 flag, controlled by the best controller found during one of the optimization runs. The target pattern is the *half-discs* pattern. Each time the variance of the energy over 16 steps is bellow 10^{-15}, a zero centered Gaussian noise with deviation 1 is added to all states of all cells, resulting in an energy spike. After such burst, the energy does not always return to the same level: The optimized controller has several attractors. In that particular example, each attractor encodes a similar pattern, which is a slight variations of the (disc) target pattern. This is not always the case, as different stable energy levels might encode for different patterns.

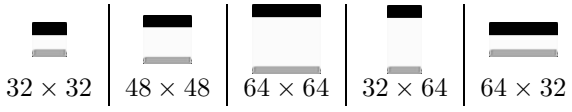


Fig. 8. The diffusion-based controller resulting from an optimization for the 32×32 *three-bands* target is used in different grid sizes. The development still features a stable attractor for all grid geometries. Those attractors fit the three bands pattern, with the correct gray levels. However, the width of the bands are not the correct ones.



Fig. 9. The diffusion-based controller resulting from an optimization for the 32×32 *disc* target is used in different grid sizes. The development still features a stable attractor for all grid geometries. Note the non-homothetic scaling of the pattern: the corners keep their round shape, while the side limits between black and white remain straight.

relevant mechanism in this setup, which was earlier left for optimization to discover. Also, reusing an optimized controller with a diffusion-based model in different flag sizes does not break that stability property. An example of final stable state in environments of varying size are shown in figures 8 and 9. Indeed, the gray levels and some features of the patterns are preserved (like the bands of the band patterns), even if the original relative proportions of the pattern shapes are lost. While this phenomenon has been observed with the NEAT-based model, the current diffusion-based model further enables an easy homothetic scaling of the pattern: Without any further optimization, a simple adjustment of the diffusion process may perform a Gaussian blur of radius 2 for a flag stretched by a factor 2.

VII. Development and Stability

In previous Sections, all proposed models relied on update function optimizers that always target development towards a steady state. Although those steady states are not always global attractors, they are at least local attractors. In this Section, the objective is both to evaluate the possible pros and cons of the *energy-based* method as well as to advocate its relevance with regard to generalization issues. This Section intends to validate the hypothesis underlying the link between the proposed halting method and the robustness issues that arise when using the experimental setup proposed in Section V-B. The goal is to study the influence of the number of development steps and of the stopping criterion on the accuracy of the results as well as on the stability of the solutions with respect to fault tolerant and self-healing.

In order to conduct these experimental studies, a set of alternative stopping criteria taken from the literature are implemented and tested against proposed the energy-based criterion. These methods are:

- The *fixed* method, that consists in setting an arbitrary number of development steps. Values 16, 32, 64, 128, 256, 512, and 1024 steps have been used. This criterion is both straight-forward and widely used (see Section III-C);
- The *multi-steps* criterion is an indirect criterion, that relies on computing the fitness value as the average score over a sliding time window (rather than one single evaluation step). The number of development steps remains however fixed (in the following: 53, 57, 62 and 63 for each patter respectively, after preliminary experiments), and the size of the averaging window for fitness evaluation is manually fixed (4 in the present experiments). The underlying idea is to enforce the accuracy of the matching during *all* considered steps, thus hopefully guiding development

towards a stable state, as originally advocated in [62] and [80].
- The *self-tuned* criterion let evolution decide of the number of developmental steps by adding it to the genotype, as in [10]. This enables evolution to adjust the development length to the level of complexity required by the target pattern;

In the following, all experiments are performed with the diffusion-based model, described in Section VI. The experimental setup is the one described in Sections V-B and VI-A: one internal and two external chemicals, and four target patterns. All results are compiled from 29 independent runs and development lasts at most 1024 steps. The previous results running the non developmental version using x/y coordinates is also featured for reference (termed "non-dev" for short).

The end of this Section details comparative results between the proposed energy-based stopping criterion and the 3 criteria listed above. Section VII-A is concerned with the accuracy of the results (i.e. matching score in the flag problem), and Section VII-B presents the results related to generalization.

A. Results: Accuracy

Figure 10 shows the matching scores of the best individual of each evolutionary run and for each setup. Results are gathered from 29 independent runs and each evolutionary run is stopped after a maximum of 10^5 evaluations. Results shown for *energy-based* correspond to the earlier *mlp-diff* variant.

The *fixed* methods are either comparable or better than the proposed *energy-based* method on all setups. Also, there appears to be a strong relationship between the development time and the median distance of the expressed pattern of the best controller to the target pattern, shorter development duration clearly leading to better performance. This is indeed a notable fact as a development of 16 steps is the lower bound for required time to create a 32×32 flag: it is the time required for chemical waves from the borders to meet at the center of the flag. Results for the *multi-steps* and *self-tuned* methods are the worst for all target patterns.

The issue of stable development is addressed in table V. This table shows, for the different stopping criteria, the number of developments that naturally end up in a stable state: whatever the initial experimental setup, all developments are allowed to go for 1024 steps (e.g. the *fixed-16* solution develops for 1024 steps unless a stable state is reached before). A first comment on these experiments is that the best performance is *never* found after the initial iteration limit, e.g., the *fixed-16* best performance is always measured before, or at, iteration 16. An example of development for the *fixed-16* criterion is provided in Figure 11. Note that the energy-based criterion is the only one whose solutions reach 100% stability. However, the *fixed* method with short development time perform relatively well, with more than half of the developments naturally terminating, except for the more difficult half-disc target. As for the rest of the runs, stability is only occasionally reached. This is true even for the *multi-steps* method, for which the stability rate seems independent of the target shape, but remains sparse (at most 50% of developments naturally terminated).

The conclusions of this series of experiments are that if the only important optimality criterion is the similarity with the target pattern, the *fixed* stopping criteria with small development times clearly perform best. Those runs reach better scores within fewer evaluations and fewer development steps, thus using much less computer resources. The *energy-based* criterion still provides the advantage of natural development termination. However, the advantage of such a feature, i.e. based on the assumption that a stable development provides a key advantage for generalization, remains to be demonstrated, and is the subject of next Section.

B. Results: Generalization

In this Section, performance on generalization is evaluated within a self-repairing scenario[2]: For each target and each criterion, the best individual from each of the 29 runs is evaluated (i.e., the development process is run) 16 times, resulting in a total of $29*16=512$ developments per method per target. The experimental setting for the developmental process is as before, except that the number of developmental steps is set to 1024, and the three first steps are strongly perturbed by a centered Gaussian noise of standard deviation 1. Perturbation occurs at the level of inputs, outputs and internal states within the whole organism, in order to start from a random position. Two complementary evaluations are conducted: the best performance after recovery disregarding stabilization and the performance after stabilization whenever it takes place (i.e. limited to runs that stabilize).

Figure 12 shows the best performances after recovery. Results shown are the best matching scores within the 1024 steps. As opposed to the pure performance setting, the longer *fixed* methods and the *energy-based* method clearly outperform all other methods.

Regarding stability, table VI shows the percentage of stable development for each method and setup. Results are close, even if slightly degraded, to what was presented earlier: the *energy-based* method still rates higher than other methods regarding stable developments. Moreover it was observed that for a given evolved solution, either *all* or *none* of the 16 experiments provided stable results, and stopped at +/- 10 steps around the same iteration as far as stable development was considered.

Lastly, performance after stabilization is considered, which of course implies only a subset of all developments. Figure 13 shows performance results after stabilization, and should be interpreted alongside the previously mentioned table VI: results are shown in Figure 13 whenever at least five candidate solutions converged to stable configuration, and the robustness of results strongly depends on the actual number of stabilized solutions. From this viewpoint, the *energy-based* method not only provides the most stable runs, as already seen above, but

[2]It is worth mentioning that generalization with respect to scalability (i.e., the ability to stabilize and generate meaningful patters with respect to some simple transformation of the original target when the size of the grid is increased) was studied in the first author's PhD [13], but is not considered to be in the scope of this study, as performance is strongly related to the definition of scalability. For example, homothetic transform is but one possible interpretation of scalability.

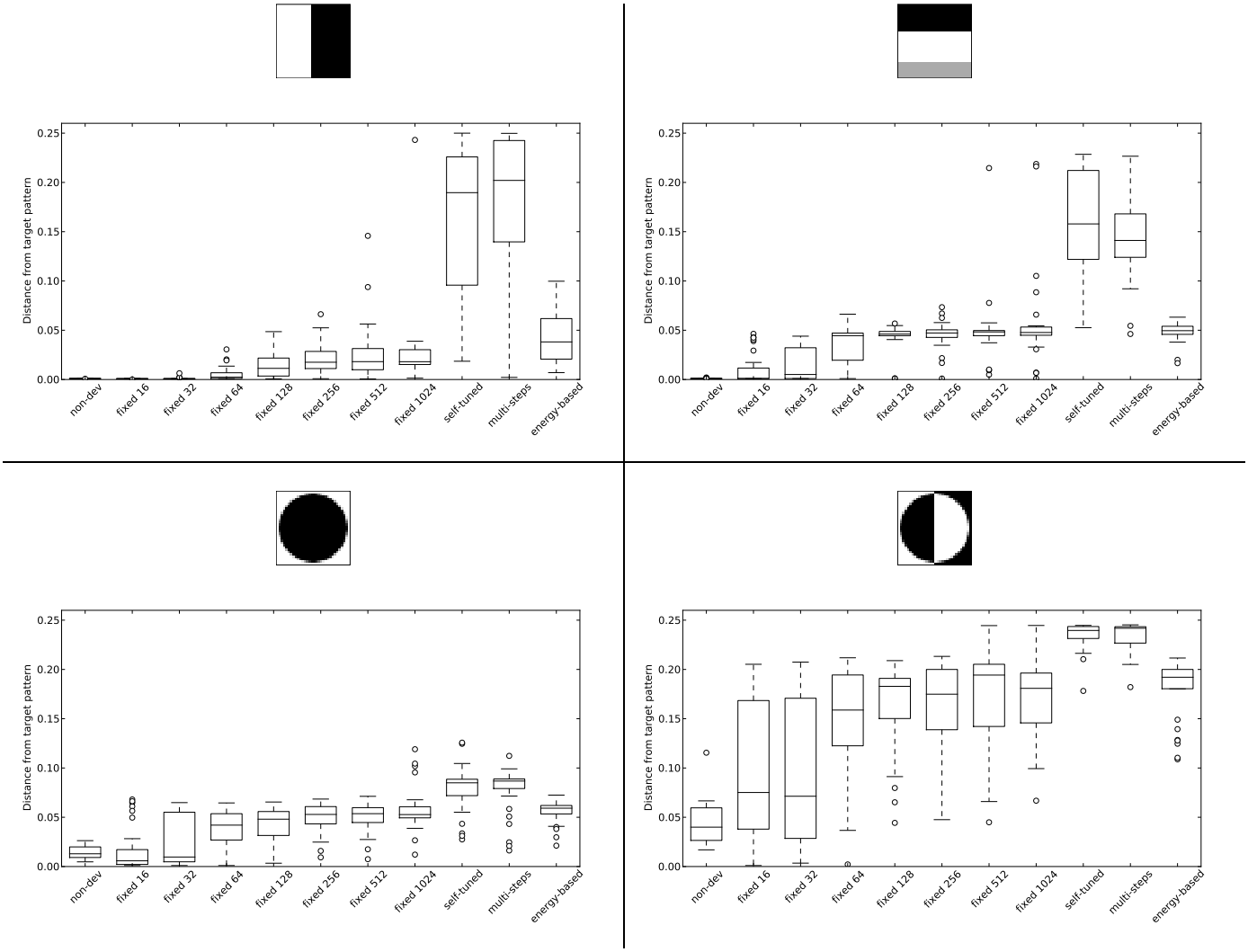


Fig. 10. Boxplots of the scores of the best solutions for various stopping critera: (*fixed*, *self-tuned* and *multi-steps*. The last column is a reminder of the results obtained when using the energy-based criterion (*mlp-diff* in previous Figures).

TABLE V

MONITORING STABLE DEVELOPMENT: NUMBER OF EMBRYOS (AMONG THE 29 SOLUTIONS FROM THE 29 INDEPENDENT RUNS) THAT STABILIZES WITHIN 1024 DEVELOPMENT STEPS FOR THE EXPERIMENTS WITH VARIOUS STOPPING CRITERIA (*fixed*, *self-tuned* AND *multi-steps*). STATISTICS ARE GATHERED USING THE BEST INDIVIDUAL OF EACH RUN. THE LAST COLUMN IS A REMINDER OF RESULTS WITH THE ENERGY-BASED CRITERION (*mlp-diff* IN PREVIOUS FIGURES).

	fixed 16	fixed 32	fixed 64	fixed 128	fixed 256	fixed 512	fixed 1024	self-tuned	multi-steps $(53, 57, 62, 63)$	energy based
two-bands	13	14	9	0	0	0	0	6	11	29
three-bands	15	13	22	8	2	0	3	4	9	29
disc	22	21	16	10	4	7	11	4	10	29
half-discs	1	4	6	4	0	1	3	8	14	29

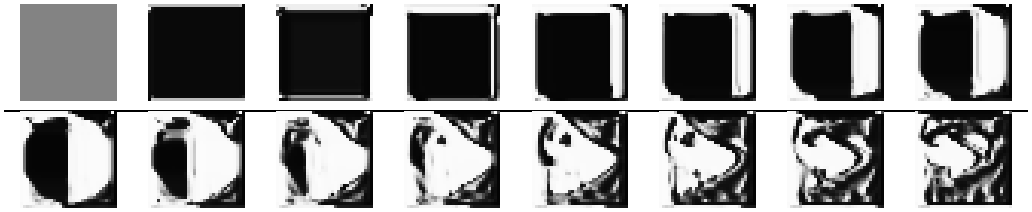


Fig. 11. Sequence of the expressed pattern of a flag controller, optimized to match the *half discs* target. Only the even steps are shown, from left to right and top to bottom. The *fixed* stopping criterion was used during evolution, limited to 16 development steps. Note the really good matching at the step 16, and the unstable dynamics at further steps.

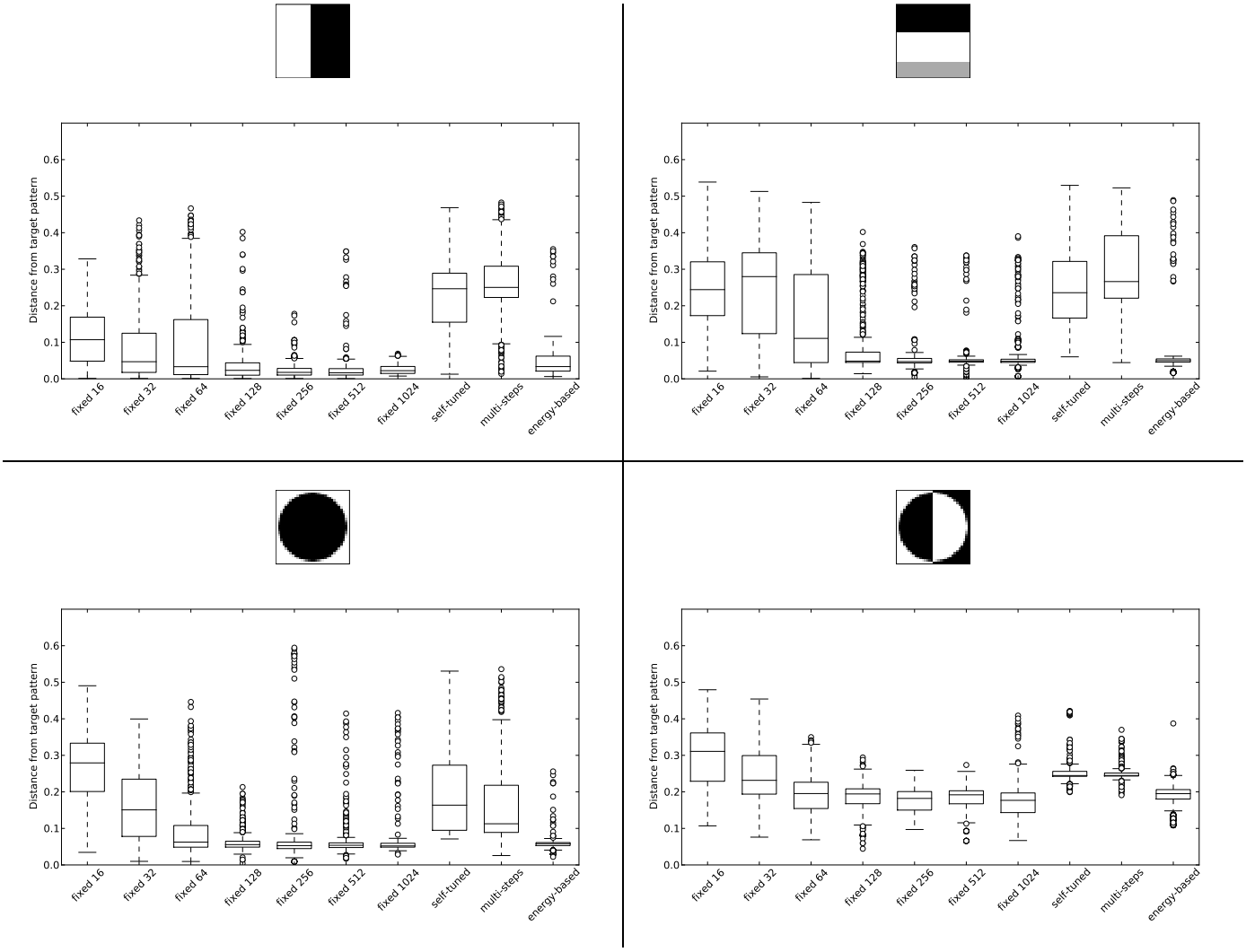


Fig. 12. Self-repairing. Results show performance for all setups considering the best match during 1024 steps. The first three iterations are perturbed with a Gaussian noise of standard deviation of 1, then development is continued until stabilization is reached (or the maximum number of allowed steps). The score for each development is the best match, and does not take into account stabilization.

TABLE VI
SELF-REPAIRING AND STABILITY. NUMBER OF EMBRYOS (AMONG THE 29 SOLUTIONS FROM THE 29 INDEPENDENT RUNS) THAT STABILIZE WITHIN 1024 DEVELOPMENT STEPS AFTER PERTURBATION. NOTE THAT NUMBERS GIVE THE AMOUNT OF *runs* THAT SUCCEEDED, NOT THE TOTAL NUMBER OF DEVELOPMENTS (I.E. ONE SUCCESSFUL RUN MEANS ALL 16 DEVELOPMENTS SUCCESSFULLY STABILIZED).

	fixed 16	fixed 32	fixed 64	fixed 128	fixed 256	fixed 512	fixed 1024	self-tuned	multi-steps $(53, 57, 62, 63)$	energy based
two-bands	13	13	8	0	0	0	0	6	10	28
three-bands	15	10	20	9	3	1	4	4	9	28
disc	20	21	17	8	4	8	10	2	9	23
half-discs	0	2	5	3	0	1	5	8	12	26

also repeatedly displays the best performance, with occasional exceptions where it is nevertheless comparable to the other approaches, despite the fact that the lack of reliable data makes these results difficult to assess (e.g. stopping criteria *fixed-64* to *fixed-1024* for the *disc* target only provide a small number of stable runs).

To conclude, while the *energy-based* method does indeed pay the price for evolving self-terminating solutions, it also ends up addressing several issues when it comes to generalization and self-repair: it provides good and competitive recovery performance and makes it possible to avoid a step-by-step evaluations to identify which (possibly transient) state represents the best solution. From a practical perspective, relying on a fixed stopping criterion that does not correspond to a stable state makes it difficult to reuse the evolved solutions in a different context: using noisy initial conditions may require an arbitrary larger number of iterations, which cannot be "guessed" a priori by the supervisor. This price may also be prohibitive depending on the problem at hand if fitness evaluations are particularly expensive (e.g. if computing the fitness requires an important amount of computation (see for example [16])). From this perspective, the *energy-based*

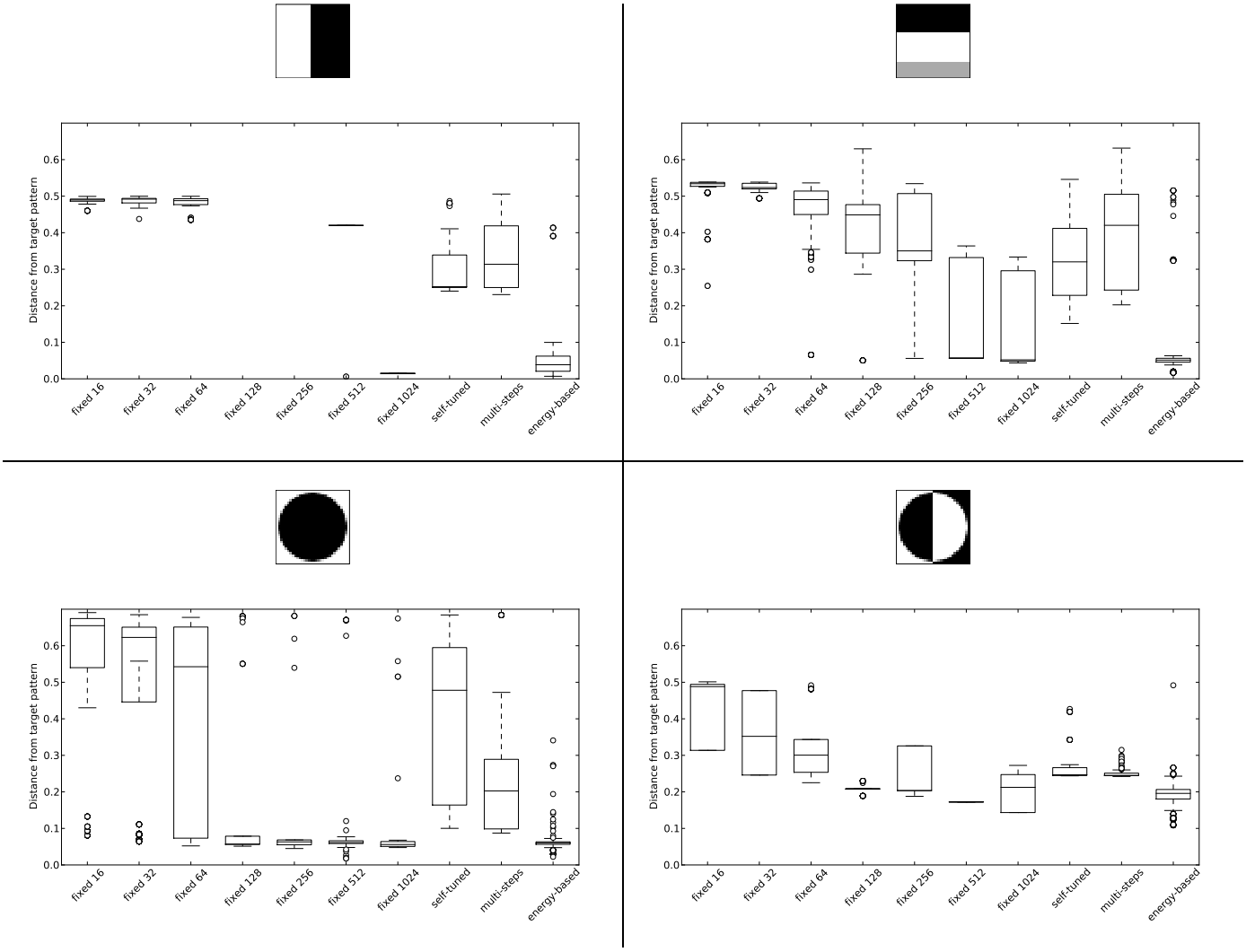


Fig. 13. Self-repairing and stability. Results show performance for all setups considering the best match after stabilization (i.e. *only run that stabilize are considered – empty data means no run stabilized. See table in figure VI for the number of successful runs*). The first three iterations are perturbed with a Gaussian noise with standard deviation of 1, then development is continued until stabilization is reached (or the maximum number of allowed steps). Only runs that stabilize are taken into account here.

method benefits from the self-termination properties without requiring any additional costly fitness evaluation.

VIII. Concluding remarks

This paper addressed the issue of robustness in multi-cellular developmental systems, claiming that the stopping criterion of the developmental process is the key to at least certain types of robustness. Our hypothesis is that there exists a strong link between robustness and the ability for a multi-cellular organism to self-terminate, i.e., reach a stable steady state without any external stopping mechanism such as an imposed upper bound on the number of development steps. The main contribution of this paper is to introduce a new stopping criterion based on monitoring the activity of the organism, and penalizing organisms that never stop developing within a large time interval.

Experiments have been conducted using various target shapes. The experimental results varied from perfect to near-perfect matches. But more significantly, *all* experiments were able to produce stable development patterns within a reasonable number of iterations with regards to the image size (requiring roughly a number of development steps equal to two or three times the number of iterations required to diffuse information from one corner to the other). In practice, individuals are selected in the early steps of the evolutionary process on their unique ability to provide stable development, then performance is gradually taken into account when almost only stable individuals remain in the population. This is a significant improvement over previous works, that imposed a fixed number of iterations as the only stopping criterion, with no guarantee to converge towards a stable state indeed.

However, the major contribution of this work regards the robustness of the resulting individuals, and the generalization capabilities of the evolved cell control architecture. The best evolved genomes have been put in different setups in order to evaluate how development occurs in such context. Two different settings have been used for these *a posteriori* experiments:

- **Self-repair**: this setup is concerned with the behavior of the development process in the presence of noise,

including the extreme case where all cells are completely randomly initialized;
- **Scalability**: this setup demonstrates the behavior of the developmental process when the update rule is used in a different geometric setting than the one it was evolved in (e.g., larger grids).

In both cases, experiments demonstrated that the proposed energy-based stopping criterion repeatedly found solutions with excellent generalization capabilities, from frequent 100% self-repair to the ability to reach both stable and qualitatively consistent patterns in a new environment.

Further experiments compared the proposed approach with different alternatives from the literature and shed light on the pros and cons of the *energy-based* criterion: enforcing self-terminating development patterns indeed has a cost with respect to pure performance when compared to some of the more straightforward methods. However, when it comes to re-using the evolved solution in different experimental conditions, such as addressing fault-tolerance, the *energy-based* criterion is shown to be both competitive with, and often even better than, the other criteria, and much easier to use, as self-termination avoids the constant monitoring of the actual performance of the developing solutions.

REFERENCES

[1] Anne Auger and Nikolaus Hansen. A restart cma evolution strategy with increasing population size. In *Proc. CEC 2005*, pages 1769–1776. IEEE Press, 2005.

[2] Rodger W. Baker and Gabor T. Herman. Celia - a cellular linear iterative array simulator. In *Proceedings of the 4th annual conference on applications of simulation*, pages 64–73, 1970.

[3] Wolfgang Banzhaf. On the dynamics of an artificial regulatory network. In W. Banzhaf et al., editor, *ECAL'03*, pages 217–227. LNAI 2801, Springer Verlag, 2003.

[4] Peter Bentley and Sanjeev Kumar. Three ways to grow designs: A comparison of embryogenies for an evolutionary design problem. In Wolfgang Banzhaf et al., editor, *GECCO'99*, pages 35–43. Morgan Kaufmann, 1999.

[5] Peter J. Bentley and Jonathan P. Wakefield. Generic evolutionary design. *Soft Computing in Engineering Design and Manufacturing*, pages 289–298, 1997.

[6] Gregory Beurier, Fabien Michel, and Jacques Ferber. A morphogenesis model for multiagent embryogeny. In *Proceedings of Artificial Life (Alife X)*, 2006.

[7] Josh Bongard. Evolving modular genetic regulatory networks. In David B. Fogel et al., editor, *CEC 2002*, pages 1872–1877. IEEE Press, 2002.

[8] Josh Bongard and Rolf Pfeifer. Evolving complete agents using artificial ontogeny. In F. Hara and R. Pfeifer, editors, *Morpho-functional Machines: The New Species (Designing Embodied Intelligence)*, pages 237–258. Springer-Verlag, Berlin, 2003.

[9] Chris P. Bowers. Simulating evolution with a computational model of embryogeny: Obtaining robustness from evolved individuals. In M. Capcarrere et al., editor, *Advances in Artificial Life*, pages 149–158. LNAI 3630, Springer Verlag, 2005.

[10] Arturo Chavoya and Yves Duthen. Using a genetic algorithm to evolve cellular automata for 2d/3d computational development. In M. Cattolico et al., editor, *Proc. GECCO'06*, pages 231–232. ACM Press, 2006.

[11] Arturo Chavoya and Yves Duthen. Use of a genetic algorithm to evolve an extended artificial regulatory network for cell pattern generation. In D. Thierens et al., editor, *Proc. GECCO'07*, pages 1062–1062. ACM Press, 2007.

[12] Yann Le Cun, John S. Denker, and Sara A. Solla. Optimal brain damage. In *Advances in Neural Information Processing Systems*, pages 598–605. Morgan Kaufmann, 1990.

[13] Alexandre Devert. *Building processes optimization: Toward an artificial ontogeny based approach*. PhD thesis, Universite Paris-Sud XI, 2009.

[14] Alexandre Devert, Nicolas Bredeche, and Marc Schoenauer. Robust multi-cellular developmental design. In D. Thierens et al., editor, *GECCO'07*, pages 982–989. ACM Press, 2007.

[15] Alexandre Devert, Nicolas Bredeche, and Marc Schoenauer. Unsupervised learning of echo state networks: A case study in artificial embryogeny. In *Artificial Evolution*, pages 278–290, 2007.

[16] Alexandre Devert, Nicolas Bredeche, and Marc Schoenauer. Artificial ontogeny for truss structure design. In *CD proceedings of the Spatial Computing SASO Workshop (SCW 2008)*, 2008.

[17] Peter Durr, Claudio Mattiussi, and Dario Floreano. Neuroevolution with analog genetic encoding. In Th. Runarsson et al., editor, *PPSN IX*, pages 671–680. LNCS 4193, Springer Verlag, 2006.

[18] Marc Ebner. Evolutionary design of objects using scene graphs. In Conor Ryan et al., editor, *Proc. EuroGP'03*, pages 47–58. LNCS 2610, Springer Verlag, 2003.

[19] Peter Eggenberger-Hotz. Evolving morphologies of simulated 3d organisms based on differential gene expression. In P. Husbands and I. Harvey, editors, *ECAL97*, pages 205–213. MIT Press, 1997.

[20] Peter Eggenberger-Hotz. Genome-physics interaction as a new concept to reduce the number of genetic parameters in artificial evolution. In *Proc. CEC'03*, pages 191–198. IEEE Press, 2003.

[21] Peter Eggenberger-Hotz. Comparing direct and developmental encoding schemes in artificial evolution a case study in evolving lens shapes. In *Proc. CEC'04*, pages 752–757. IEEE Press, 2004.

[22] Nicolás S. Estévez and Hod Lipson. Dynamical blueprints: exploiting levels of system-environment interaction. In D. Thierens et al., editor, *Proc. GECCO'07*, pages 238–244. ACM Press, 2007.

[23] Diego Federici. Using embryonic stages to increase the evolvability of development. In J. Miller, editor, *Proc. GECCO Workshop on Regeneration and Learning in Developmental Systems, WORLDS 2004*. ACM Press, 2004.

[24] Diego Federici and Tom Ziemke. Why are evolved developing organisms also fault-tolerant? In S. Nolfi et al., editor, *From Animals to Animats, Proc. SAB'06*, pages 449–460. LNCS 4095, Springer Verlag, 2006.

[25] Nicholas Flann, Jing Hu, Mayank Bansal, Vinay Patel, and Greg Podgorski. Biological development of cell patterns: Characterizing the space of cell chemistry genetic regulatory networks. In M. Capcarrere et al., editor, *Advances in Artificial Life*, pages 57–66. LNAI 3630, Springer Verlag, 2005.

[26] John Frazer. *An Evolutionary Architecture*. Architectural Association Publications, http://www.aaschool.ac.uk/publications/ea/intro.html, 1995.

[27] Pablo Funes and Jordan Pollack. Evolutionary body building: Adaptive physical designs for robots. *Artif. Life*, 4(4):337–357, 1998.

[28] Pablo Funes and Jordan B. Pollack. Computer evolution of buildable objects. In P. Husbands and I. Harvey, editors, *Fourth European Conf. on Artificial Life*, pages 358–367, Cambridge, MA, 1997. MIT Press.

[29] Timothy G. W. Gordon and Peter J. Bentley. Bias and scalability in evolutionary development. In H.-G. Beyer et al., editor, *Proc. GECCO'05*, pages 83–90. ACM Press, 2005.

[30] Timothy G.W. Gordon and Peter J. Bentley. Development brings scalability to hardware evolution. In Jason Lohn et al., editor, *Proc. NASA/DoD Conf. on Evolvable Hardware*, pages 272–279. IEEE Computer Society, 2005.

[31] Garrison W. Greenwood and Andrew M. Tyrrell. *Introduction to Evolvable Hardware: A Practical Guide for Designing Self-Adaptive Systems*. Wiley-IEEE Press, 2006.

[32] Frederic Gruau. Automatic definition of modular neural networks. *Adaptive Behavior*, 3(2):151–183, 1994.

[33] Frederic Gruau. *Neural Network Synthesis using Cellular Encoding and the Genetic Algorithm*. PhD thesis, Ecole Normale Superieure de Lyon, France, 1994.

[34] Frederic Gruau, Darrell Whitley, and Larry Pyeatt. A comparison between cellular encoding and direct encoding for genetic neural networks. In John R. Koza et al., editor, *Genetic Programming 1996*, pages 81–89, Stanford University, CA, USA, 1996. MIT Press.

[35] Hatem Hamda, Francois Jouve, Evelyne Lutton, Marc Schoenauer, and Michele Sebag. Compact unstructured representations in evolutionary topological optimum design. *Applied Intelligence*, 16:139–155, 2002.

[36] Hatem Hamda and Marc Schoenauer. Adaptive techniques for evolutionary topological optimum design. In I. Parmee, editor, *Evolutionary Design and Manufacture*, pages 121–132. Springer Verlag, 2000.

[37] Gabor T. Herman and Wu-Hang Liu. The daughter of celia, the french flag and the firing squad. In *WSC '73 : Proceedings of the 6th conference on Winter simulation*, page 870, 1973.

[38] Gregory S. Hornby. *Generative Representations for Evolutionary Design Automation*. PhD thesis, Brandeis University, Dept. of Computer Science, Boston, MA, USA, February 2003.

[39] Gregory S. Hornby. Measuring, enabling and comparing modularity, regularity and hierarchy in evolutionary design. In H.-G. Beyer et al., editor, *Proc. GECCO'05*, pages 1729–1736. ACM Press, 2005.

[40] Gregory S. Hornby and Jordan B. Pollack. Evolving *L*-systems to generate virtual creatures. *Computers and Graphics*, 25(6):1041–1048, 2001.

[41] Kurt Hornik, Maxwell Stinchcombe, and Halbert White. Multilayer feedforward networks are universal approximators. *Neural Networks*, 2:359–366, 1989.

[42] Rafal Kicinger, Tomasz Arciszewski, and Kenneth A. De Jong. Morphogenesis and structural design: cellular automata representations of steel structures in tall buildings. In *Proceedings of the Congress on Evolutionary Computation (CEC'2004)*, pages 411–418, Piscataway, NJ, 6 2004. IEEE Press.

[43] Rafal Kicinger, Tomasz Arciszewski, and Kenneth D. Jong. Evolutionary computation and structural design: A survey of the state-of-the-art. *Computers & Structures*, 83(23-24):1943–1978, 2005.

[44] Rafal Kicinger, Tomasz Arciszewski, and Kenneth De Jong. Parameterized versus generative representations in structural design: an empirical comparison. In H.-G. Beyer et al., editor, *Proc. GECCO'05*, pages 2007–2014. ACM Press, 2005.

[45] Jerome Kodjabachian and Jean-Arcady Meyer. Evolution and development of neural controllers for locomotion, gradient-following, and obstacle-avoidance in artificial insects. *IEEE Transactions on Neural Networks*, 9:796–812, 1998.

[46] Taras Kowaliw, Peter Grogono, and Nawwaf Kharma. Environment as a spatial constraint on the growth of structural form. In D. Thierens et al., editor, *Proc. GECCO'07*, pages 1037–1044. ACM Press, 2007.

[47] John R. Koza, Forrest H. Bennet, David Andre, and Martin A. Keane. *Genetic Programming III: Automatic Synthesis of Analog Circuits*. MIT Press, Massachussetts, 1999.

[48] Peter Krcah. Evolutionary development of robotic organisms. Master's thesis, Faculty of Mathematics and Physics, Department of Software and Computer Science Education, Charles University, Czech Republic, 2007.

[49] Sanjeev Kumar. A developmental genetics-inspired approach to robot control. In F. Rotlauf, editor, *Proc. GECCO'05 Workshops*, pages 304–309. ACM Press, 2005.

[50] Paul Dwight Kuo, Andre Leier, and Wolfgang Banzhaf. Evolving dynamics in an artificial regulatory network model. In Xin Yao et al., editor, *Proc. PPSN'04*, pages 571–580. LNCS 3242, Springer Verlag, 2004.

[51] Nicolas Lassabe, Herve Luga, and Yves Duthen. New skills for evolving artificial creatures. *International IEEE Journal of Information Technology and Intelligent Computing*, 1:120–146, 2008.

[52] Aristid Lindenmayer and Grzegorz Rozenberg. Developmental systems and languages. In *proceedings of the 4th annual ACM symposium on theory of computing (STOC'72)*, pages 214–221, 1972.

[53] Hod Lipson. Principles of modularity, regularity, and hierarchy for scalable systems. *Journal of Biological Physics and Chemistry*, 7(4):125–128, 2007.

[54] Hod Lipson and Jordan B. Pollack. Automatic design and manufacture of robotic lifeforms. *Nature*, 406(6799):974–978, August 2000.

[55] Jason Lohn, James Crawford, Al Globus, Gregory S. Hornby, William Kraus, Gregory Larchev, Anna Pryor, and Deepak Srivastava. Evolvable systems for space applications. In *Proc. Intl Conf. on Space Mission Challenges for Information Technology*, 2003.

[56] Jason Lohn, William F. Kraus, Derek S. Linden, and Silvano Colombano. Evolutionary optimization of yagi-uda antennas. In Y. Liu et al., editor, *Proc. ICES '01: From Biology to Hardware*, pages 236–243, London, UK, 2001. Springer-Verlag.

[57] Steven Manos, Maryanne C. J. Large, and Leon Poladian. Evolutionary design of single-mode microstructured polymer optical fibres using an artificial embryogeny representation. In D. Thierens et al., editor, *Proc. GECCO'07*, pages 2549–2556. ACM Press, 2007.

[58] Claudio Mattiussi, Daniel Marbach, Peter Dürr, and Dario Floreano. The Age of Analog Networks. *AI Magazine*, 29(3):63–76, 2008.

[59] Thomas Miconi and Alastair Channon. An improved system for artificial creatures evolution. In *in Procs 10th Intl Conf on Simulation and Synthesis of Living Systems (ALIFE*, pages 255–261. MIT Press, 2006.

[60] Julian F. Miller. Evolving developmental programs for adaptation, morphogenesis and self-repair. In W. Banzhaf et al., editor, *Proc. ECAL'03*, pages 256–265. LNCS 2801, Springer Verlag, 2003.

[61] Julian F. Miller. Evolving a self-repairing, self-regulating, french flag organism. In *Proc. GECCO'04*, pages 129–139. LNCS 3102 and 3103, Springer Verlag, 2004.

[62] Julian F. Miller and Wolfgang Banzhaf. Evolving the program for a cell: from french flags to boolean circuits. In R. Nagpal, editor, *On Growth, Form and Computers*, pages 278–301. Academic Press, 2003.

[63] Miguel Nicolau and Marc Schoenauer. Evolving specific network statistical properties using a gene regulatory network model. In G. Raidl et al., editor, *Proc. GECCO'09*, pages 723–730. ACM Press, 2009.

[64] Stefano Nolfi and Domenico Parisi. Growing neural networks. Technical Report PCIA-91- 15, Institute of Psychology, CNR, Rome, 1991.

[65] Una-May O'Reilly, Mark Hemberg, and A. Menges. Evolutionary computation and artificial life in architecture. *Architectural Design: Special Issue on Emergence*, 2004.

[66] Chandana Paul, Hod Lipson, and Francisco J. Valero Cuevas. Evolutionary form-finding of tensegrity structures. In H.-G. Beyer et al., editor, *Proc. GECCO'05*, pages 3–10. ACM Press, 2005.

[67] Pavel Petrovic. Solving lego brick layout problem using evolutionary algorithms. In *Proc. Norsk Informatik Konferanse (NIK)*, pages 87–97, 2001.

[68] Maxim Peysakhov. Representation and evolution of lego-based assemblies. In C. Ryan, U.-M. O'Reilly, and W.B. Langdon, editors, *GECCO Graduate Student Workshop*, pages 297–300, Las Vegas, Nevada, USA, 2000.

[69] Maxim Peysakhov and William C. Regli. Using assembly representations to enable evolutionary design of lego structures. *Artificial Intelligence for Engineering, Design, Analysis and Manufacturing*, 17(2):155–168, 2003.

[70] Juraj Plavcan and Pavel Petrovic. Direct and indirect representations for evolutionary design of objects. In D. Thierens et al., editor, *Proc. GECCO'07*, pages 160–171. ACM Press, 2007.

[71] Jordan B. Pollack and Hod Lipson. The golem project: Evolving hardware bodies and brains. In *Proc. 2nd NASA/DoD workshop on Evolvable Hardware, EH'00*, pages 37–42. IEEE Computer Society, 2000.

[72] Torsten Reil. Dynamics of gene expression in an artificial genome - implications for biological and artificial ontogeny. In D. Floreano et al., editor, *Proc. ECAL'05*, pages 457–466. LNCS 1674, Springer Verlag, 1999.

[73] Joseph Reisinger and Risto Miikkulainen. Acquiring evolvability through adaptive representations. In D. Thierens et al., editor, *Proc. GECCO'07*, pages 1045–1052. ACM Press, 2007.

[74] Karl Sims. Evolving virtual creatures. In *SIGGRAPH'94*, pages 15–22. ACM Press, July 1994.

[75] Kenneth O. Stanley. *Efficient evolution of neural networks through complexification*. PhD thesis, The University of Texas at Austin, 2004.

[76] Kenneth O. Stanley. Compositional pattern producing networks: A novel abstraction of development. *Genetic Programming and Evolvable Machines: Special Issue on Developmental Systems*, 8(2):131–162, 2007.

[77] Kenneth O. Stanley, David B. D'Ambrosio, and Jason Gauci. A hypercube-based indirect encoding for evolving large-scale neural networks. *Artificial Life*, 15(2):185–212, 2009.

[78] Kenneth O. Stanley and Risto Miikkulainen. Evolving neural networks through augmenting topologies. *Evolutionary Computation*, 10(2):99–127, 2002.

[79] Kenneth O. Stanley and Risto Miikkulainen. A taxonomy for artificial embryogeny. *Artificial Life*, 9(2):93–130, 2003.

[80] Till Steiner, Lisa Schramm, Yaochu Jin, and Bernhard Sendhoff. Emergence of feedback in artificial gene regulatory networks. In *Proc. CEC'07*, pages 867–874. IEEE Press, 2007.

[81] Till Steiner, Jens Trommler, Martin Brenn, Yaochu Jin, and Bernhard Sendhoff. Global shape with morphogen gradients and motile polarized cells. In *Proc. CEC'09*, pages 2225–2232. IEEE Press, 2009.

[82] Gunnar Tufte and Pauline C. Haddow. Towards development on a silicon-based cellular computing machine. *Natural Computing*, 4(4):387–416, 2005.

[83] Alan M. Turing. The chemical basis of morphogenesis. *Royal Society of London Philosophical Transactions Series B*, 237:37–72, 1952.

[84] Lewis Wolpert. The french flag problem: a contribution to the discussion on pattern development and regulation. In C. Waddington, editor, *Towards a Theoretical Biology*, pages 125–133. Edinburgh University Press, 1968.

[85] Lewis Wolpert. Positional information and the spatial pattern of cellular differentiation. *Journal of Theoretical Biology*, 25:1–47, 1969.

[86] Lewis Wolpert. *Principles of Development*. Oxford University Press, 2006.