Biomimetic Engineering. *Acta Hort.* (ISHS) **802**:21-32, 2008.

Biomimetic Engineering

Francisco J. Vico

Grupo de Estudios en Biomimética
ETS Ingeniería Informática, Universidad de Málaga
Parque Tecnológico de Andalucía (Málaga, Spain)
fjv@geb.uma.es

1. Mother Earth's computing report

Humankind is a privileged animal species for many reasons. A remarkable one is its ability to conceive and manufacture objects. Human industry is indeed leading the various winning strategies (along with language and culture) that has permitted this primate to extraordinarily increase its life expectancy and proliferation rate. (It is indeed so successful, that it now threatens the whole planet.) The design of this industry kicks off in the brain, a computing machine particularly good at storing, recognizing and associating patterns. Even in a time when human beings tend to populate non-natural, man-made environments, the many forms, colorings, textures and behaviors of nature continuously excite our senses and blend in our thoughts, even more deeply during childhood. Then, it would be exaggerated to say that Biomimetics is a brand new strategy. As long as human creation is based on previously acquired knowledge and experiences, it is not surprising that engineering, the arts, and any form of expression, is influenced by nature's way to some extent.

The design of human industry has evolved from very simple tools, to complex engineering devices. Nature has always provided us with a rich catalog of excellent materials and inspiring designs. Now, equipped with new machinery and techniques, we look again at Nature. We aim at mimicking not only its best products, but also its design principles.

Organic life, as we know it, is indeed a vast pool of diversity. Living matter inhabits almost every corner of the terrestrial ecosphere. From warm open-air ecosystems to the extreme conditions of hot salt ponds, living cells have found ways to metabolize the sources of energy, and get organized in complex organisms of specialized tissues and organs that adapt themselves to the environment, and can modify the environment to their own needs as well. Life on Earth has evolved such a diverse portfolio of species that the number of designs, mechanisms and strategies that can actually be abstracted is astonishing. As August Krogh put it: "For a large number of problems there will be some animal of choice, on which it can be most conveniently studied".

The scientific method starts with a meticulous observation of natural phenomena, and humans are particularly good at that game. In principle, the aim of science is to understand the physical world, but an observer's mind can behave either as an engineer or as a scientist. The minute examination of the many living forms that surround us has led to the understanding of new organizational principles, some of which can be imported in our production processes. In practice, bio-inspiration can arise at very different levels of observation: be it social organization, the shape of an organism, the structure and functioning of organs, tissular composition, cellular form and behavior, or the detailed structure of molecules. Our direct experience of the wide portfolio of species found in nature, and their particular organs, have clearly favored that the initial models would come from the organism and organ levels. But the development of new techniques (on one hand to observe the micro- and nanostructure of living beings, and on the other to simulate the complex behavior of social communities) have significantly extended the domain of interest.

The biggest computer program

Abiogenesis, the problem of the origin of life in our planet, is an important question for science (and fundamental to Astrobiology) where different disciplines can contribute. Whether panspermic or emergent from a non-living substrate, what seems to be clear is that life has been evolving on Earth for, at least, half the time since it was formed (between 4.4 and 2.7 billion years). In a way, the planet could be seen as a colossal parallel computer where computation is performed by interacting atomic particles.

Assuming that life had an endogenetic origin, early metabolism and replication were but two special types of reactions among the myriads of possible combinations of elements. The point is that somehow they happened, and finally combined to form the precursors of current cells. Artificial chemistry is a subfield of Artificial Life that tries to reproduce a computational variety of these phenomena. Experiments like *Tierra*, *Avida* and *Amoeba* simulated the evolution of very simple computer programs, and accounted for the emergence of life-characteristic processes, like parasitism, immunity, symbiosis, and (in the case of *Amoeba*) spontaneous replication.

This is very impressive, considering the limited computational power that these experiments involved. But a much more difficult problem would be to develop systems with the structural and functional complexity of bacteria, not to speak of the intricate design of organisms made of billions of interacting eukaryotic cells. Obviously, our computing machinery has little to do with the huge infrastructure of a planetary lab. Just to give an idea, considering only the small percentage of carbon atoms of the evolving biomass, and their atomic interactions at the femtoseconds scale, a lower bound for the terrestrial computation power can be estimated in $5 \cdot 10^{40}$ PFLOPS[1], and since the beginning of life this little portion of organic living matter would have completed at least $3 \cdot 10^{55}$ PFLOPs. Since the most powerful computers nowadays can compute at a rate of half a PFLOPS, even by using a room filled with 1,000 of them, $2 \cdot 10^{45}$ years (more than 10^{35} times the age of the universe!) would be necessary to match this small fraction of biomass evolution. We, the engineers, can of course rely on our computers to support the design process, or even to automate and optimize parts of it, but we can also start by learning how similar problems were treated by mother Earth computer during the last million years.

In order to give an idea of the influence that nature is having on our current (and future) technology, a number of case studies have been selected and grouped into three categories: designs, methodologies, and artificial life. We will start by analyzing the most popular set of examples: those that mimic living structure and function.

2. Mimicking nature’s designs

Biomimetics is generally identified with a group of successful stories developed in the field of new materials. Velcro, a fastener inspired by the bur’s hooks, is probably the most popular and ubiquitous case (Figure 1). But the many examples that have arisen in recent years show that we are only starting to understand the potential of this strategy. In order to give a broad idea of how diverse and powerful can be nature’s inspiration, a number of case-studies are exposed next.

[1] PFLOPS stands for peta-floating point operations per second, i.e. 10^{15} basic numerical computations performed in a second.

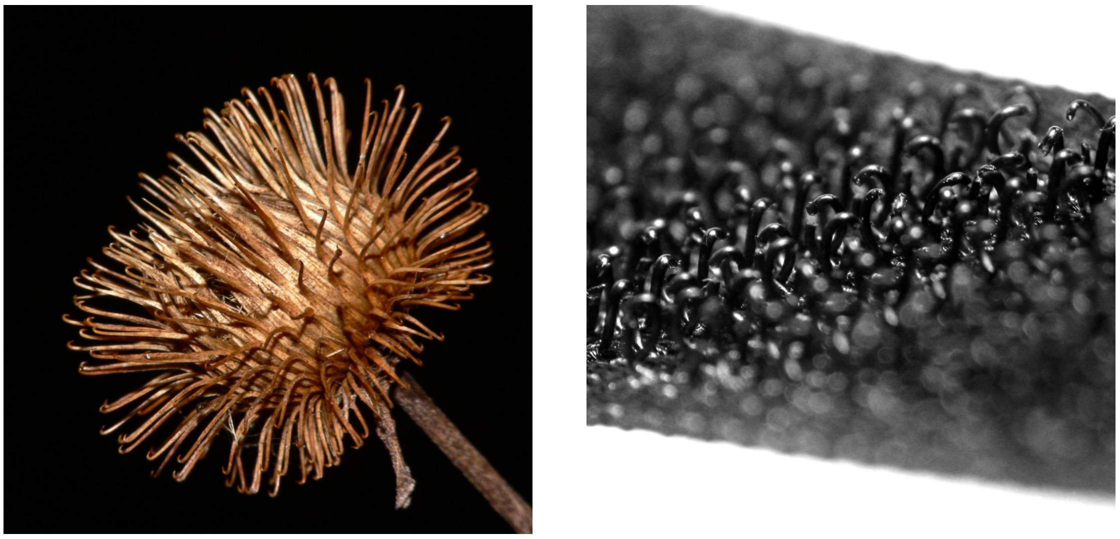

Figure 1. The most successful biomimetic design: Velcro. A bur covered by hooks (left, 🅭 Brad Smith) inspired the design principle of one the most ubiquitous fastener of our days (right, 🅭 Andrew Magill).

Mercedes-Benz's concept vehicle is a state-of-the-art example of how a multidisciplinary team can mix bio-inspired ideas and design constraints together. Among all the different possible choices of animals, the team focused on the underwater world, and in particularly the boxfish. Because it inhabits the coral reefs, this fish must protect itself from collisions, and move around in limited spaces, also being quick to escape from predators. All these features made it a good candidate to inspire a car's design. Figure 2 shows the streamlined boxfish and the final vehicle, with a *Cd* value (a measure of aerodynamic drag) of 0.19, it is among the most aerodynamic cars in its category.

Reptiles: an animal group that has shown a high adaptability to very different environments, it is a good source of inspiration for engineers. The gecko, a lizard that can walk on vertical surfaces, has made us think of a new line of gravity-defying robots. *Stickybot*, developed by Mark Cutkosky and Sangbae Kim at Stanford University, is based on the directional adhesion of the gecko's toes, and can actually climb glass, acrylic and whiteboard smooth surfaces. In Figure 2, a gecko and the quadruped robot are shown. Applications of *Stickybot* point towards surveillance and lifesaving.

Another example can be taken from plants' leaves. The cleaning properties of a leaf's microstructure (the lotus effect) inspired the *GrennShield* fabric finish, and *Lotusan* paint (Figure 2), products that creates water and stain repellency on textiles. These inventions reproduce, on a given surface, the roughened microstructure of leaves, which minimizes the adhesive force of water. In this way, water rolls of the surface, without getting stuck to it. Additionally, dirt particles are absorbed by the water droplets, what results in an efficient mechanism to clean surfaces.

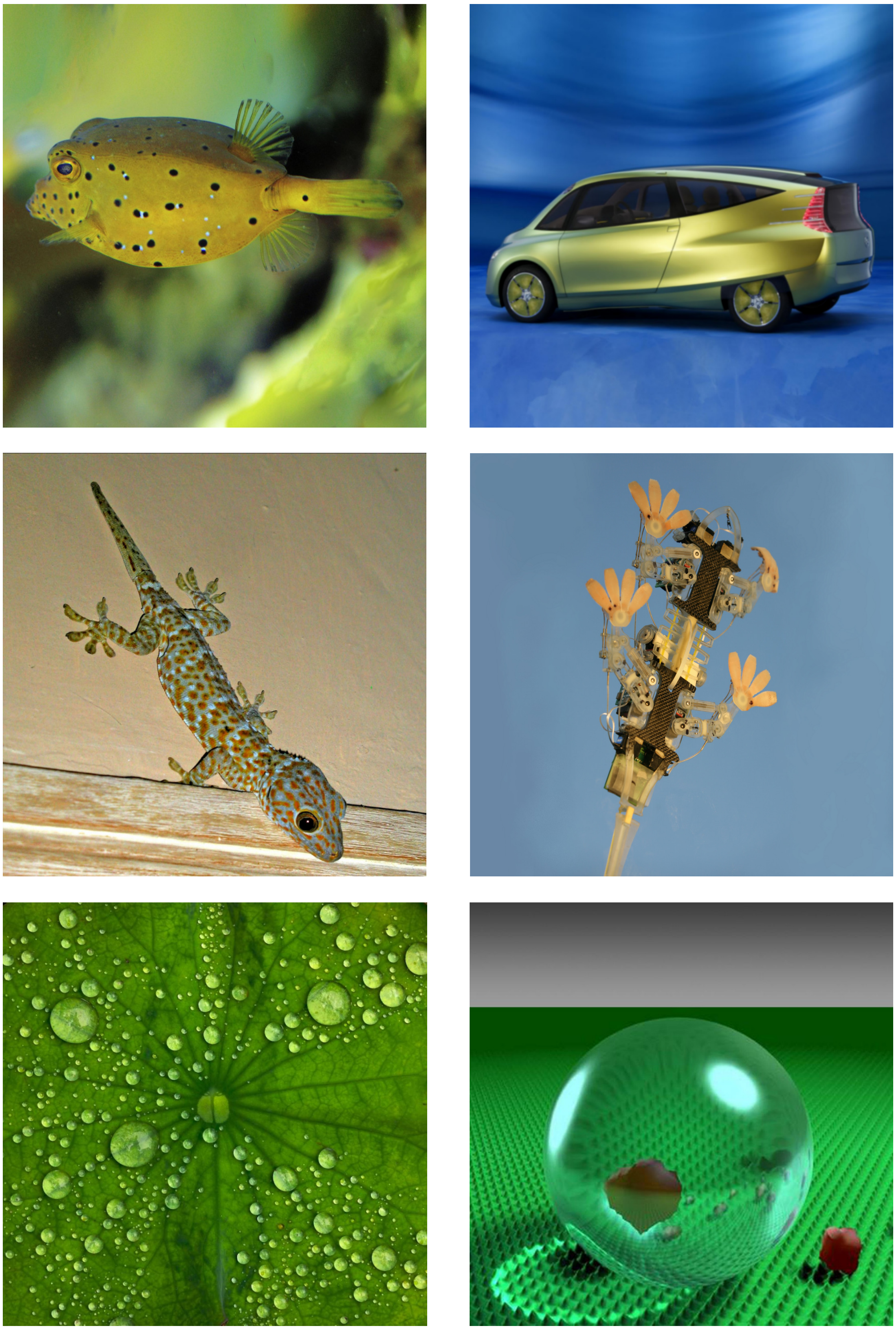

Figure 2. First row: Mercedes-Benz's bet in biomimetic technology. The boxfish (left, © Daimler AG) shows an angular design, but nonetheless very streamlined. The resulting concept vehicle (right, © *Daimler AG*) is one of the most aerodynamic compact cars. Second row: the gecko (left, 🅭 Autan), a lizard that can walk on vertical surfaces, was the inspiring lizard for *Stickybot* (right, © Mark Cutkosky, Standford University), a robot that mimics its behavior with dry adhesive toes. Third row: a leaf's surface (left, 🅭 uBookworm) showing the lotus effect: water droplets that do not adhere to the surface. *Lotusan*® (right, © STO) is a paint that reproduces this effect by generating a similar microstructure. Particles on such a surface tend to stick to droplets.

3. Mimicking nature’s strategies

Besides the fitted living designs exposed in the previous section, there are other secrets that nature maintains hidden. Engineers produce amazing materials and machines inspired by Biology, but new algorithms and computational paradigms can also be conceived based on our current knowledge of nature. Here, we analyze the motivation and properties of the three main computational paradigms: neuromorphic, swarm-based, and evolutionary. However, this is only a representative sample of the variety of processes that are currently abstracted from life.

Neural computation

The nervous system is a tissue responsible for the control of the organism in some living beings. The brain is a highly connected network of cells that (1) perceives the self body and the outer world, (2) massively processes information, and (3) adapts the organism to the environment by activating appropriate behavioral patterns. Although the brain is a quite inaccessible organ to experimental methodologies, we are starting to uncover its general organization, internal patterns of connectivity, and the different ways in which neurons can represent concepts and take decisions.

One of the lines of work in the field of Artificial Intelligence is inspired by the physiology of the neural cell, and the distributed approach of the brain to computing functions. Artificial neural networks constitute a paradigm of distributed computation that integrates different combinations of neuron models, strategies of connectivity, learning rules, and activation dynamics. Its main contributions have been to supply engineers with methods that adjust themselves to the characteristics of the data, and to provide a common representation to a large variety of data processing techniques, some of them already known. Neural networks have a wide range of applications: in problems like function approximation, optimization, clustering and classification.

An increasing computer power has logically led to the simulation of more complex neural models and more biologically realistic networks. After the efforts initiated at the end of the last century, Computational Neuroscience is now ready to propose new paradigms based on the structure and functioning of the mammalian brain (Figure 3). A new generation of learning rules has demonstrated that the final receptive fields (the way neurons connect to others) emerge as a consequence of the stimulation received by the neuron, and its interaction with neighboring cells. Structures characteristic of the primary visual cortex have now been obtained *in silico*. Neurons processing thousands of connections, and networks of several thousands of nerve cells, are currently simulated to understand the dynamics of a cortical column (a processing element at the cortex level).

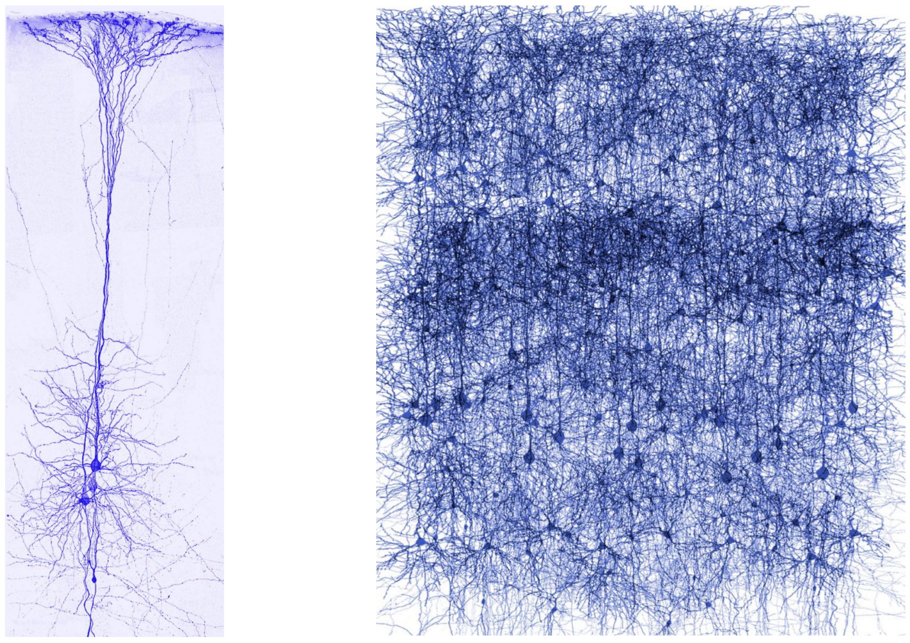
Figure 3. The complexity of a pyramidal neuron (left), and a snapshot of the cortical microcircuitry of the mammalian brain (right). © IBM Research.

One of their main properties of artificial neural networks (as the ability to learn and self-organize) is also the main argument of their detractors: after learning, neural nets indeed do the job, but the acquired knowledge is distributed in the connection matrix, which is unreadable for the engineer. Perhaps this is the fate of most bio-inspired computing models: the more realistic and powerful they are, the less we understand how they actually solve the problems.

Swarm intelligence

The collective behavior of social insects and animals has also attracted the attention of scientists. Despite the simplicity of the rules that govern the actions of an ant, a honeybee, or a herring, when arranged into colonies, swarms or schools show an emergent intelligent behavior. Computing and robotics can benefit from this phenomenon, if they think in a distributed way: pools of moving particles that change their position according to neighboring ones, or small robots that communicate with others to accomplish a global task. This paradigm has been termed “agent-based approach”, since it consists in programming the simple behavior of cooperating agents.

Swarm behavior has inspired algorithms that are currently applied in job scheduling tasks, regulation of air traffic, and routing trucks.

Particle swarm optimization

The flocking behavior of some birds was one of the first modeling efforts which showed that emergent properties can be obtained from the interaction of simple agents. Craig Reynolds, a computer graphics researcher, simulated clouds of agents that tried to fly in the average direction of their neighbors, and stay close to them whilst not crowding them (Reynolds, 1987). These three simple rules, when operating on all the agents, nicely reproduced the expected flocks.

Inspired by this phenomenon, particle swarm optimization (PSO) is an amazingly effective optimization method (Kennedy & Eberhart, 1995). The performance of PSO has been tested on landscapes where the global optimum is not easy to reach because it is surrounded by similar local optima (Figure 4 shows a number of particles converging to the global solution).

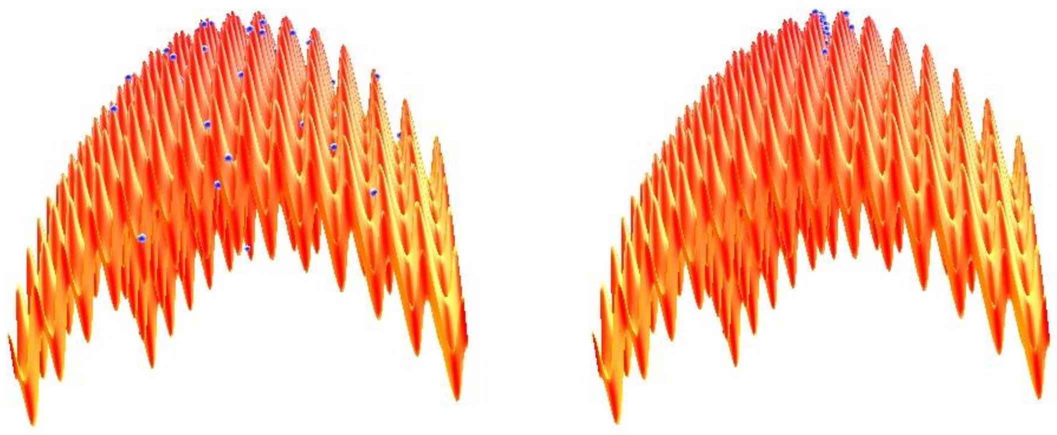

Figure 4. Particle swarm optimization of the non-convex Rosenbrock function. The global optimum is surrounded by local optima. The initial population of particles (left, blue spheres) rapidly forms a cluster in the neighborhood of the global optimum (right), and finally all particles concentrates in this optimum.

Ant colony optimization

Ant colonies constitute a remarkable example of swarm intelligence. Ants do not actually see nearby neighbors, but they can smell their mates, and leave messages by modifying their environment (stigmergy). Interacting in a simple way ant colonies develop efficient strategies. They can implement, for example, job scheduling: the colony (not a particular scheduler) decides how many ants will forage, and which ants will be dedicated to maintenance. Finding short paths between two points is another problem ant colonies are good at. By trying different itineraries, and leaving pheromonal trails on their way, ants find out the best routes to communicate the nest with food sources. Figure 5 illustrates the application of the ant colony optimization (ACO) algorithm (Colorni *et al.*, 1992) to a traditional NP-hard[2] problem: the traveling salesman problem (TSP).

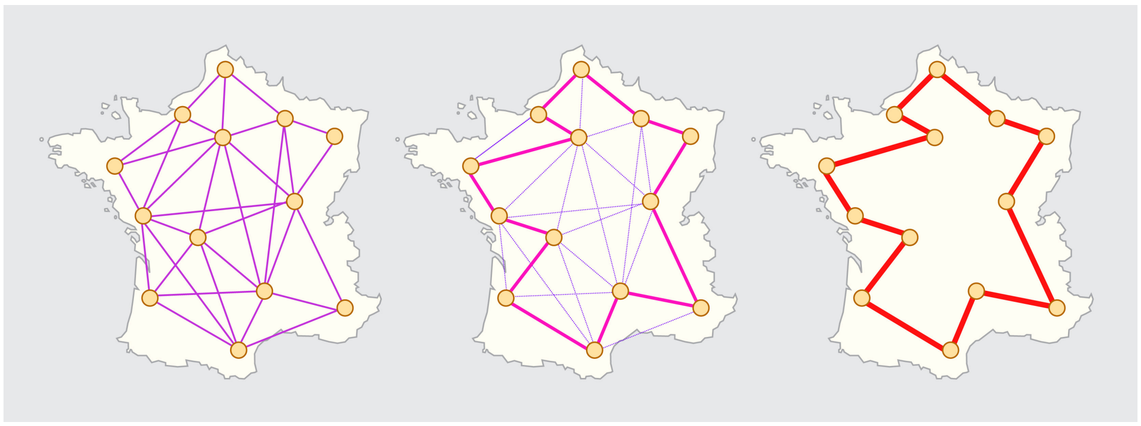

Figure 5. An example of TSP solved with ACO. Simulated ants follow alternative itineraries leaving a trail behind them (left). Most promising paths are visited more often (center). Ants finally chose a single route, and alternative paths are discarded (right). 🅭 Johann Dréo.

Simulated ants visit all the cities and then start the way back to the nest, leaving pheromones with an intensity that is inversely proportional to the length of the path traveled. At each node, ants decide where to go based on the pheromonal information on each track. Since pheromonal information vanishes with time, this simple interaction allows the colony to establish the optimal path.

Evolutionary computation

But if engineers can copy living designs and their behaviors, they can also copy the general principle that originates them all: biological evolution.

The field of evolutionary computation agglutinates a number of methods inspired in the many different strategies brought into play in nature to evolve successful species. Genetic mutation, sexual recombination and natural selection are amongst the most popular strategies in evolutionary and genetic algorithms (Holland, 1975). Not as well-known, but just as efficient, are some other operators, like speciation, niching, crowding, elitism, and a list that grows with our gaining knowledge in genetics.

Traditionally included in the soft-computing framework, evolutionary computation is a general approach to efficiently explore a huge solutions space, with little analysis and programming effort. In principle, any optimization problem can be attacked with this

[2] In the NP-hard TSP, the number of possible solutions increases exponentially with the number of cities.

approach, if a number of conditions are met: (1) the objective can be expressed in a function that estimates the fitness of any solution, (2) the structure of the solution is coded in the genome, (3) a population of solutions is randomly generated, and (4) successive generations are obtained by changing and mixing the most fitted individuals. Natural selection operates on the population, making it converge to optimum solutions.

That is what explains the popularity of genetic algorithms: setup and simulation times are generally short, and after some parameter tuning, good solutions are reached. Applications of evolutionary computation have been reported in basically any engineering field, from VLSI layout generation, to electric load forecasting. Figure 6 refers to one of the first applications developed in industrial design: the evolution of aesthetic and ergonomic telephone handsets. This example has an additional interest in this context because it hybridized two bio-inspired techniques: the fitness function was approximated by an artificial neural network (Vico *et al*. 1999).

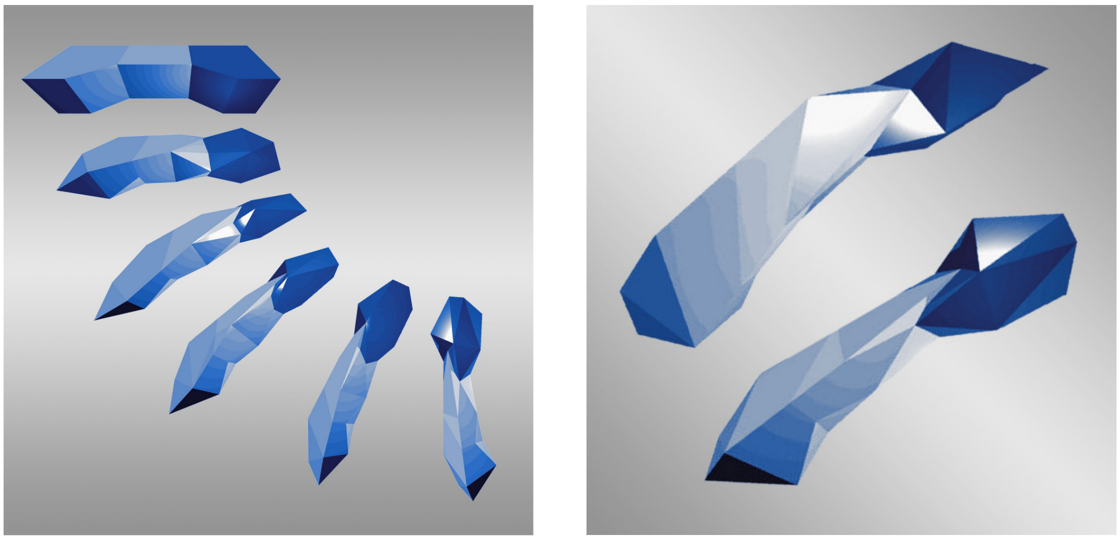

Figure 6. Evolution of a telephone handset (left): from an initial blocky design (upper-left corner), a genetic algorithm sculpts the shape in search of a design that meets the aesthetic and ergonomic constraints. The right figure shows two views of the final design. © GEB, Universidad de Málaga.

But the success of this technique has found a limit. When the nature of the solution is structured (i.e. there are strong dependences among the genes that represent it), modifying a good solution tends to disrupt it, and the population experiments very little progress, if any. Some additional constraints and operators are needed in this case.

Genetic programming

One case of evolving structured solutions that has received special attention is the automatic generation of programs that meet concrete specifications, also known as genetic programming (Koza, 1992). A sequential program is usually represented as a list of lines, each containing a single instruction. Given a concrete program (usually written in LISP language), the probability of making a modification that transforms it into a valid program is extremely low: any symbol, or even instruction, that is changed is most likely to introduce syntactic errors.

Computer programs, like natural language sentences, can be derived from a generative grammar. Adding this constraint to the evolutionary search, the solutions (programs) of a concrete problem are represented as trees, where the dependencies among control structures, instructions and operators are clearly expressed. This representation preserves the viability of the solutions, and strongly influences the design of the genetic operators. Mutation and recombination, for example, are made according to the information represented in particular nodes and subtrees, respectively.

The impact of genetic programming has not gone too far. Perhaps the main contribution of this paradigm was to attract our attention to more elaborated ways of representing the information. A new line in this direction focuses on the development process of higher organisms.

The embryonic approach

Multicellularity was a keystone in the evolution of complex organisms. A mammal, a bird, or a reptile is made of billions of specialized eukaryotic cells, arranged into organs with an intricate structure and function. In contrast to protists, the general organization (or bodyplan) of these organisms undergoes a developmental stage where cells proliferate from a single one by bipartition. This process, known as the embryonic development of a living being, is the readout of the information contained in the genetic material.

The beginning of this century has seen an explosion of activity in genomics, proteomics, and the life sciences in general, and we seem committed into understanding the dynamics of eukaryotic genetic expression. Inspired by this principle, some algorithms are renewing the field of evolutionary computation by introducing an implicit strategy of information encoding, opposed to the traditional explicit encoding of solutions. This approach brings potential computational capabilities, but also a few new problems. Traditional genetic operators, for example, must be redesigned, since the disruption introduced by mutation and recombination of solutions is huge. An algorithm to develop the solution (phenotype) from the encoded information (genotype) is also needed here.

The overall performance will be very much influenced by the properties of this process, and how well it fits with the genetic operators.

This new approach is only just starting to receive attention, and the interest of most applications is more in their bio-inspired strategy, rather than in their true performance.

A developmental approach has also been used to design the structure of a landing vehicle that could be used for planetary exploration. Figure 7 shows the landing strategy used by the most fitted structure. The genetic search included a genetic expression model based on the execution of simple graph-rewriting rules, and a physically-plausible simulation of the landing. The fitness function measured the displacement of the lander from the impact point, and the alterations suffered by the structure. The best vehicle was found to develop radial symmetry, and organic footpads.

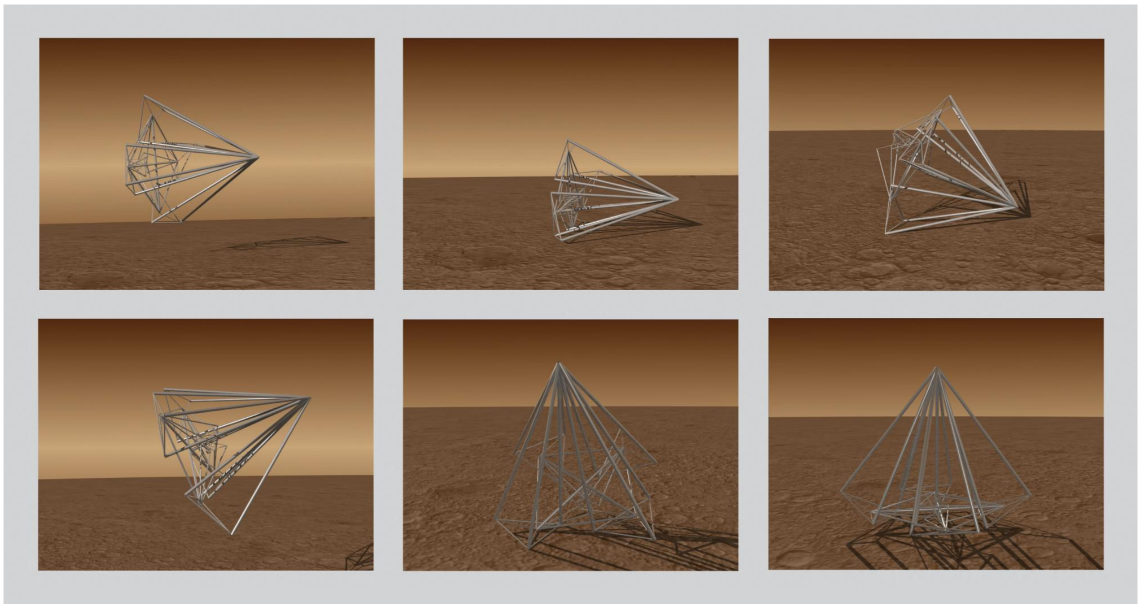
Figure 7. Artist rendition of a landing sequence. The structure of the best-fitted lander better absorbs the impact (top), and rests very close to the impact point (bottom). 🅭 GEB, Universidad de Málaga.

Evolutionary computation is gradually leaving the traditional single-processor implementation. Computer clusters, grid computing, and supercomputers will exploit the power of its implicit parallelism in the next few years to come. New biomimetic evolutionary algorithms (and complex tasks fitted to their computational properties) will be designed to exploit the extra computer power that is now available.

Robotics

The field of robotics has been traditionally influenced by the structures and behavior of living forms. From humanoid artifacts to the more ground-breaking artificial insects that now invade robotic labs, engineers have abstracted models of locomotion, orientation, perception, and other skills for a basic interaction with the environment.

However, current bio-inspired robotics are moving in a different direction: instead of reproducing animal body parts, engineers are focusing on more abstract life attributes, like self-reproduction, resiliency, or embodiment. Figure 8 shows two representative examples of this new-wave: self-reproducing machinery (Zykov *et al*. 2005), and a legged robot that self-models its body, i.e. learns an internal model of itself by interacting with the environment (Bongard *et al*. 2006).

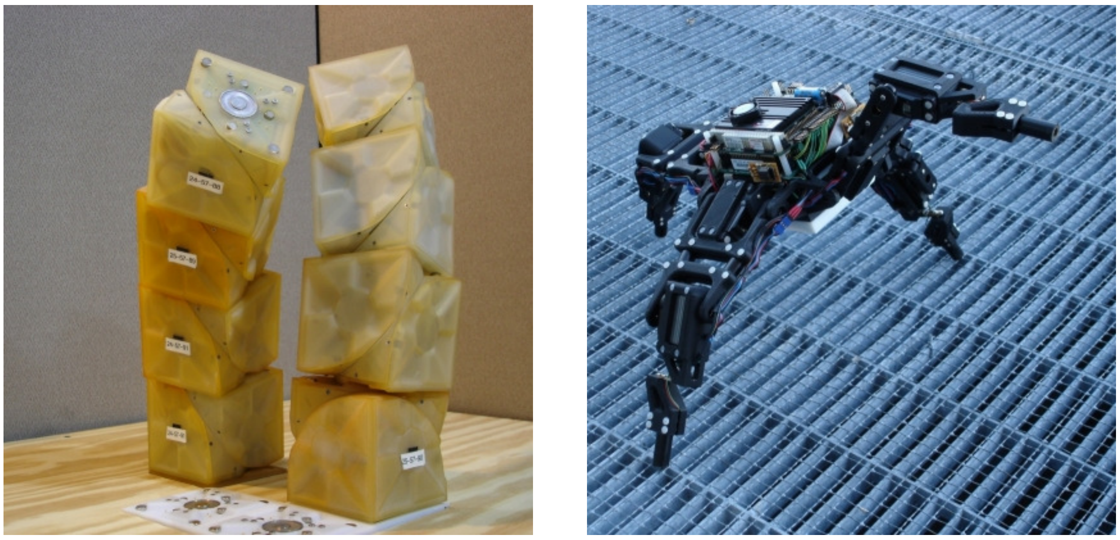
Figure 8. Two examples of biomimetic robots: self-reproducing 3D four-module robot (left), and self-modeling legged robot (right). Courtesy of Cornell University (NY).

4. Mimicking life itself

In addition to copying nature's designs, and mimicking its strategies, current technology has given us the tools to take up yet again an old human dream: to create life. This is perhaps now a much more attainable goal, not only because we have learned some key principles of life organization (like the complexity of the nucleic acids machinery), but because the concept of life itself is being revisited. The emerging field of Artificial Life (AL) is based on the assumption that "... the 'logical form' of an organism can be separated from its material basis of construction" (Langton, 1989). In this sense, we are starting to differentiate the traditional interpretation of *life-as-we-know-it* from the more general concept of *life-as-it-could-be*.

Although the concept of AL has been given different interpretations, the "strong" variant claims that AL can produce living entities. More precisely (as researcher Tom

Ray put it) a system can be considered alive if it is capable of some basic properties of life, like self-replication and open-ended evolution. Leaving apart the discussion of whether AL products do actually live or not, the simulation of living processes is generating knowledge and new techniques that approach the complexity of organic life. Cellular automata, Boolean networks, and L-systems are good examples of this mathematical portfolio.

Among cellular automata, John Conway's *Game of Life* (Gardner, 1970) is, by far, the best-known example. Simulated on an unlimited discrete lattice of binary cells, the value of each cell is synchronously updated according to the values of neighboring cells. The updating rule is the same for each cell, but what many people find surprising is that very simple rules generate high levels of complexity. The *Game of Life* simply turns on or off cells according to the number of neighbors (the eight adjacent cells) that are active on the previous state. Running this rule on a lattice, a repertoire of self-organizing structures emerges. These objects have been given names in accordance to their behavior: still-life, oscillators, gliders, spaceships, etc. The emergence of these patterns from randomness is a key question in the theoretical study of life: rules like this are said to be balanced, in the sense that keeps the activation in between global activation and collapse, a regime that has been termed "the edge of chaos". Figure 9 illustrates this phenomenon with a pattern that oscillates shifting to the right, while it leaves behind a collection of pseudorandom structures. This trial is not ordered, nor chaotic.

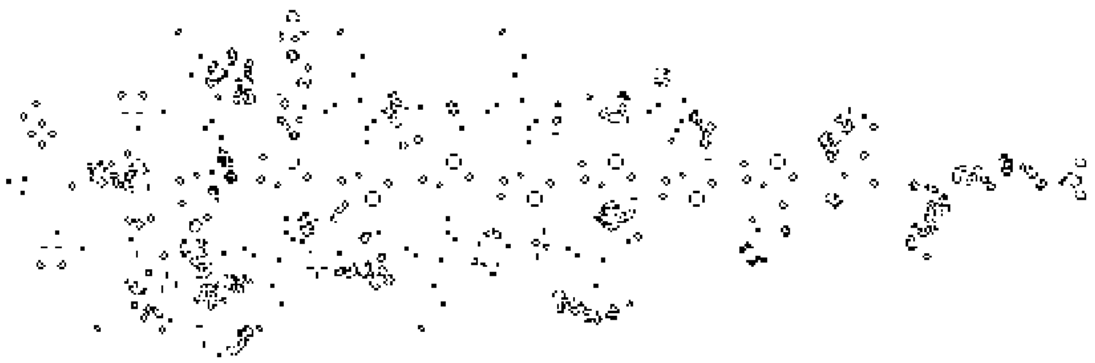

Figure 9. A complex pattern generated by the *Game of Life*: a puffer (cells at the right) is a structure that moves in a single direction (rightwards, in this picture), leaving behind a trail of pseudorandom patterns.

A vast number of rules have been studied, and the variety of results is impressive. The series of triangular patterns observed on the shell of some mollusks, and the waves of the Belousov-Zhabotinsky reaction were nicely replicated by 1D and 2D cellular automata, respectively (Figure 10).

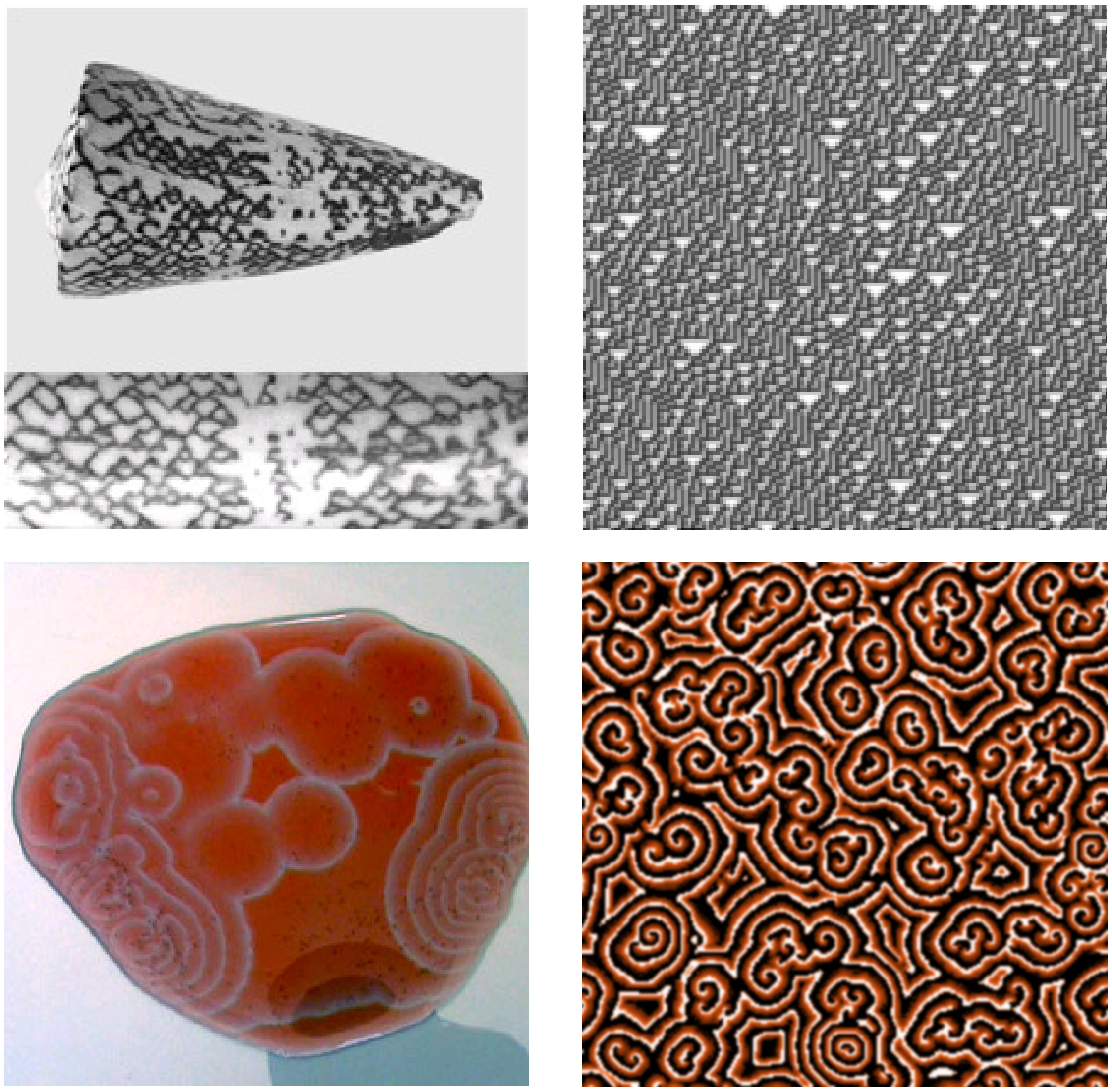

Figure 10. First row: the triangular patterning of a mollusk's shell (left) and similar traces left by a 1D cellular automaton as it develops in time (right). © Stephen Wolfram, LLC. Second row: Belousov-Zhabotinsky reaction (left), and a reproduction of this wavy behavior on a 2D cellular automaton (right).

A different modeling approach based on grammatical systems is the L-system proposed by Aristide Lindenmayer to model the growth of plants, from algae to branching higher plants (Prusinkiewicz & Lindenmayer, 1990). Despite this biology-oriented motivation, L-systems have found application in many different fields, ranging from graphics to architecture (Figure 11).

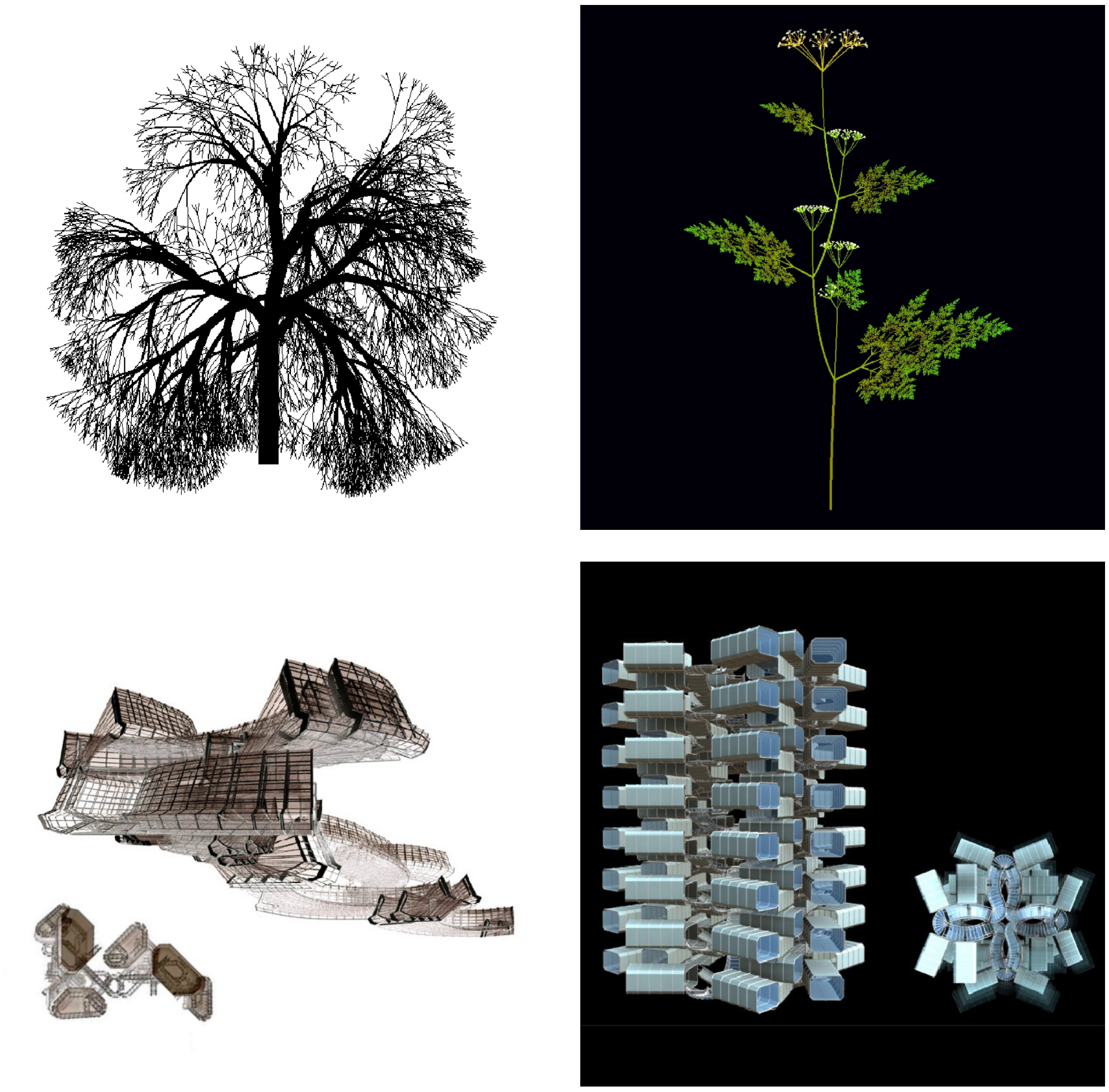

Figure 11. First row: modeling of plants with L-systems (Prusinkiewicz & Lindenmayer, 1990). Second row: two building layouts generated by L-systems. © Michael Hansmeyer.

Cellular automata and L-systems are formal systems. Their initial motivation was centered on the mathematical analysis of their properties. The computer simulation of these models has taught us much about their dynamic behavior, and has spread out the usage and potential application to different fields. Much more complex models of living beings and behavior can be depicted now in the computer era. Tensegric systems have been recently proposed as the basis of living matter at very different levels (Ingber, 1998). Eukaryotic cellular architecture (microfilaments and microtubules) and tissular organization (e.g. bony tissue) are just two examples of living tensegrities. Understanding and applying this principle for balancing tension and compression components demand much more computational power than the previous models, but its potential to build self-stabilized structures is huge. Deployable structures, mobile robots, and multicellular models (Figure 12) have been proposed based on this biological strategy.

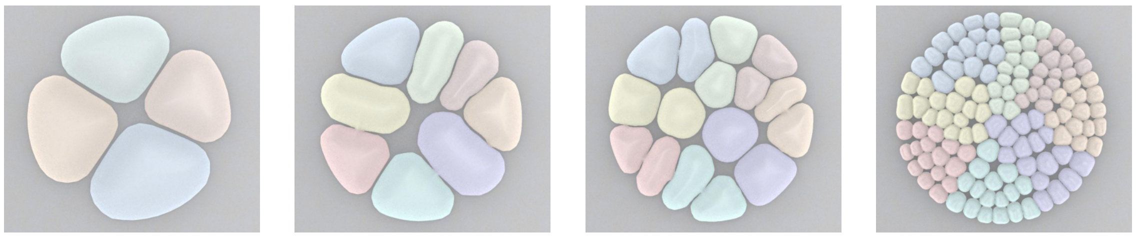

Figure 12. Towards the development of a multicellular organism. Cell components (membrane and cytoskeleton) are modeled as a network of springs. Mitotic division and genetic expression are also embedded in the model. Artist rendition of the organism after 2, 3, 4 and 7 division (from left to right) with linage identified by colors after the third division. 🅭 GEB, Universidad de Málaga.

5. Concluding remarks

Nowadays, engineering is being very much influenced by the observation of nature. As we have seen, the observation of living solutions is not the only source of inspiration. The way nature combines solutions, the complex interactions within populations of cells or living beings, the general principles of life organization, and the simulation of life itself, provide powerful strategies that can be imported into engineering as well.

We are just entering the Century of Biology, but it is not difficult to think of a future of engineering tightly bound to computer science, mathematics, and an emerging biology.

Meanwhile, it is our responsibility to preserve and keep working this planetary computer we are living in, full of life and inspiration.

6. Acknowledgments

The author is forever grateful for all the help given by the GEB staff/ members (Daniel, José David, Gema, Vicente and Ellen); through their stimulating discussions, graphic and linguistic help and not to forget their faith in a biomimetic future.

References

Bongard J., Zykov V. & Lipson H. (2006) Resilient machines through continuous self-modeling. *Science*, **314**(5802):1118-1121.

Colorny A., Dorigo M & Maniezzo V. (1992) Distributed optimization by ant colonies, In F.J. Varela & P. Bourgine (Eds.), *Proceedings of the First European Conference on Artificial Life*, 134-142. Cambridge, MA: MIT Press.

Gardner, M. (1970) Mathematical Games: The fantastic combinations of John Conway's new solitaire game "Life". *Scientific American*, **223**: 120–123.

Holland, J.H. (1975) *Adaptation in natural and artificial systems*. University of Michigan Press, Ann Arbor.

Ingber, D.E. (1998) The architecture of life. *Scientific American Magazine*, January.

Kauffman, S.A. (1969) Metabolic stability and epigenesis in randomly constructed genetic nets. *Journal of Theoretical Biology*, **22**:437-467.

Kennedy J. & Eberhart R.C. (1995) Particle swarm optimization. In *Proceedings of IEEE International Conference on Neural Networks*, **IV**:1942-1948.

Koza, J.R. (1992) *Genetic programming: on the programming of computers by means of natural selection*. Cambridge, MA: MIT Press.

Langton, C. G. (Ed.) (1989). *Artificial life*. Reading, MA: Addison-Wesley.

Prusinkiewicz, P. & Lindenmayer, A. (1990) *The algorithmic beauty of plants*. Springer-Verlag.

Reynolds C.W. (1987) Flocks, herds, and schools: A distributed behavioral model. *ACM SIGGRAPH Computer graphics*, **21**(4):25-34.

Vico F, Veredas F, Bravo JM & Almaraz J (1999) Automatic design synthesis with artificial intelligence techniques. *Artificial Intelligence in Engineering*, **13**(3):251-256.

Zykov V., Mytilinaios E., Adams B. & Lipson H. (2005) Self-reproducing machines. *Nature*, **435**(7038):163-164.