Co-evolution of Structures and Controllers for Neubot Underwater Modular Robots

Barthélémy von Haller[1], Auke Ijspeert[1], and Dario Floreano[2]

[1] Biologically Inspired Robotics Group
[2] Laboratory of Intelligent Systems,
Swiss Federal Institute of Technology (EPFL), Lausanne, Switzerland
{barthelemy.vonhaller, auke.ijspeert, dario.floreano}@epfl.ch
http://birg.epfl.ch, http://lis.epfl.ch

Abstract. This article presents the first results of a project in underwater modular robotics, called Neubots. The goals of the projects are to explore, following Von Neumann's ideas, potential mechanisms underlying self-organization and self-replication. We briefly explain the design features of the module units. We then present simulation results of the artificial co-evolution of body structures and neural controllers for locomotion. The neural controllers are inspired from the central pattern generators underlying locomotion in vertebrate animals. They are composed of multiple neural oscillators which are connected together by a specific type of coupling called synaptic spreading. The co-evolution of body and controller leads to interesting robots capable of efficient swimming. Interesting features of the neural controllers include the possibility to modulate the speed of locomotion by varying simple input signals, the robustness against perturbations, and the distributed nature of the controllers which makes them well suited for modular robotics.

1 Introduction

This article presents an adaptive scheme for underwater navigation of modular robots based on the artificial co-evolution of body structures and neural controllers for locomotion. The modular robots used in this paper are part of a long-term project [1] aimed at conceiving robots capable of self-construction and self-reproduction, as first proposed by John von Neumann. Therefore, we call such robots *Neubots* from von NEUmann roBOTS, which also means New Robots in German.

The neural controllers studied in this paper are central pattern generators (CPGs), which are networks capable of producing coordinated patterns of rhythmic activity while being modulated by simple non-oscillatory signals. Our CPGs represent an interesting framework for modular robotics because of (1) their distributed nature, (2) the locality of their interactions, (3) their robustness against perturbations, and (4) their ability of coordinating multiple degrees of freedom while being modulated by simple input signals.

In this article, the controllers are designed incrementally, with first the design of a neural oscillator which serves as the building block for the complete CPGs.

In a second stage complete CPGs are co-evolved with the body structure. During this second stage, the neural oscillators embedded into each robot unit are coupled to oscillators in neighboring units using synaptic spreading, which corresponds to projecting connections between two types of neurons to the same type of neurons in neighbor neural oscillators. This type of intercoupling is well-suited for modular robotics because of its locality and because it can be described in fewer parameters than an all-to-all coupling between oscillators.

Neubots and Related Modular Robots. More than 50 years ago, John von Neumann investigated the possibility of designing physical robots that can self-assemble and self-reproduce. He arrived to the conclusion that such robots should be composed of a dozen different types of simple modules produced in hundreds of thousands of copies [3]. Von Neumann also argued that the control system of such modules should be composed of some sort of McCulloch-Pitts neurons [4]. However, von Neumann abandoned this line of research because of technological limitations of that time and concentrated on the computational aspects of such systems, which eventually resulted on the birth of cellular automata. We think that today's technology and science of complex self-organizing systems is ripe for the realization of physical self-assembling and self-reproducing robots. In the Neubots project, which is described more extensively elsewhere [1], we continue from where von Neumann left and redefine some of his intuitions in light of recent scientific and technological advancements.

The Neubot project rests on three main principles: 1) an heterogeneous and large pool of simple and specialized modules that can be combined in various ways to form a multicellular artificial organism; 2) a set of mechatronic and control mechanisms that allows the active recruitment and release of modules by the growing and self-reproducing organism; 3) a process of intrinsic and open-ended evolution of the organism mediated by its cells (modules), which possess the entire genome of the organism. In order to simplify the recruitment and release of modules, we conceived the early prototypes as underwater units. In this paper we investigate candidate solutions for simple navigation of the Neubot modules and we focus our investigation only on one specific module, which operates as a joint. The modules are made of faceted hull with 6 actuators (Figure 1). The connectors are based on a magnetic system composed of one permanent magnet and three yokes. A motor allows switching connectors between attractive, repulsive, and neutral states. Physical prototypes of the module have been constructed and tested [2], but only simulation experiments will be presented in this article.

Neubots belong to the larger family of self-organizing modular robots. In particular, the Neubot modules described here are similar to the M-Tran and Hydron robotic systems. The M-Tran system is composed of several identical modules that can connect to each other by means of active magnetic surface [6]. Robots made of such components can autonomously transform themselves into different shapes and use the joints to walk. However, the detaching process requires a lot of energy and is rather slow. The Hydron robotic system instead is composed of several identical waterproof and spherical units that are suspended
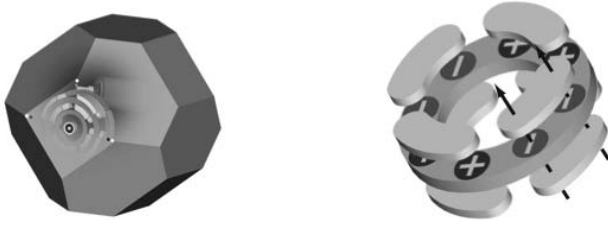
Fig. 1. Left: Schematic view of a Neubot's module and its actuation. **Right**: Schematic view of the connector.

in water and can move by means of water jets. These robots are currently used for exploring principles of self-organization [7], but cannot connect to each other. The Neubot modules used here are underwater units that can connect to each other by means of active magnets that require considerably less energy to detach and attach. Furthermore, the contact interface of the Neubot modules can rotate, thus permitting a complex articulation of the entire system.

Adaptive Locomotion for Modular Robots. A promising way to control locomotion is provided by the *Central Pattern Generators* (CPGs) observed in invertebrate and vertebrate animals [10].

CPGs have been used in the modular M-Tran system described above [11,12], in "monolithic" robots [13], and in simulated robots [14]. CPGs are interesting for modular robotics because of their distributed nature and their ability to generate efficient locomotion for complex multi-DOF structures while being modulated by simple control signals. One of the novelties of our approach is the use of neural oscillators connected by synaptic spreading (see next sections). The approach is well suited for a distributed implementation and for the optimization by a genetic algorithm.

In this paper we will co-evolve the body structures and the locomotion controllers of the modular robots in a three-dimensional simulation. Other examples of co-evolution include Karl Sims seminal work [8], Framsticks, a three-dimensional simulation project which offers various genotypes and fitness functions, to co-evolve morphology and control of virtual stick creatures [15], the work by Pollack and colleagues [16], and other interesting projects [17]. Our approach differs from previous work mainly in the type of neural controllers that we use (see above).

2   Co-evolving Structures and Controllers for Locomotion

2.1   Neuronal Simulation of the CPG, Evolution of Oscillators

The CPG models are designed incrementally. In a first stage, we evolve a canonical neural oscillator, which produces stable oscillations and whose frequency can be adapted by simple tonic signals. In a second stage, we co-evolve the body structure and the neural controller of multi-unit robots.

As in [14], neural oscillators are evolved from neural networks which have a left-right symmetry (cf Fig. 2) with three neurons of different "type" (A, B and C) and one motoneuron (M) on each side. Each neuron receives a tonic (i.e. non-oscillating) input BS representing the driving signals produced by the brain stem in vertebrates.

Neuron model. The neuron units are modeled as leaky integrators. According to this model, the mean membrane potential m_i and the short-term average firing frequency x_i of a neuron i are governed by the equations:

$$\tau_i \frac{dm_i}{dt} = -m_i + \sum_j w_{i,j} x_j \qquad \text{and} \qquad x_i = (1 + e^{-(m_i + b_i)})^{-1}$$

where b_i is the neuron's bias, τ_i is a time constant associated with the passive properties of the neuron's membrane, and $w_{i,j}$ is the synaptic weight of a connection from neuron j to neuron i. Both the neuron parameters and the synaptic weights are evolved.

Genetic Algorithm. GALib [20] was used for the real-number genetic algorithm (GA). The GA begins with a population of $N = 100$ randomly created individuals. At each generation, crossover, mutation and pruning operators are applied for creating $C = 50$ new children. For parent selection we used a rank-based roulette wheel method which chooses a parent with a probability which is inversely proportional to its rank. The crossover operator takes two parents and for each position exchanges the genes at that position with probability $P_C = 0.5$. Mutation changes, with a probability $P_M = 0.4$, the real value of a gene according to a Gaussian distribution (SD = 1.0) around the old value. The pruning operator is specific to the neural network optimization and prunes a connection (probability $P_P = 0.05$) by setting the gene corresponding to the weight of this connection to zero. The children are then evaluated and, as the population size is fixed, the $C = 50$ worst individuals are rejected from the total population (parents and children). The GA is stopped when the difference between the current best-of-generation score and the one 10 generations back in time is less than 1 percent.

Encoding. The parameters of a neural network are encoded into chromosomes which are fixed-length strings of real values. The genome encodes both the neural parameters of each neuron type —the time constant (τ), the bias (β), and the sign (excitatory (+) or inhibitory (−))— and the connectivity of the network — the synaptic weights of the outwards connections from itself to other neurons (including self-connections) and the synaptic weights from the tonic drive (BS, left and right). The motoneurons M are forced to be excitatory and do not have connections to other neurons. The total number of genes is 43.

Fitness Function. The fitness function is defined to reward solutions that (1) oscillate regularly with left and right motoneurons out-of-phase, (2) have as few connections as possible, and (3) have a frequency of oscillation that can be varied monotonically with the level of BS. Mathematically, the fitness function is a product of six factors:

$$fit = nb\_connect \cdot oscil \cdot regularity \cdot oscil\_phase \cdot freq\_range \cdot ampl$$
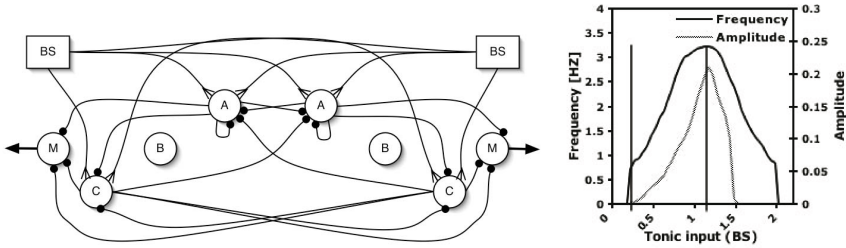
Fig. 2. Left: The neuronal oscillator. A black point indicates an inhibitory connection, a fork indicates an excitatory connection. The neurons of type B are not connected to any other neuron and can therefore be removed from the network. **Right**: Evolution of the frequency and amplitude when varying the input drive (BS). The vertical lines determine the region in which the amplitude and frequency increase monotonically.

where $nb\_connect$ corresponds to the inverse of the number of connections from a neuron, $oscil$ to the number of optima (in order to favor oscillating networks), $regularity$ to the inverse of the Standard Deviation of periods, $oscil\_phase$ to $|\theta_{oscil} - 0.5|$ with θ_{oscil} the phase between the left and right motoneuron, $freq\_range$ to max_freq/min_freq with $min\_freq \geq 0.8$ and finally $ampl$ corresponds to the mean of amplitude. Each factor varies linearly between, and is bounded by, 0.05 and 1.0.

The network is evaluated during 6 seconds after a stabilization time of 6 seconds. The original input (BS) is set arbitrary to 1.0, but the network is evaluated multiple times with small BS increments and decrements in order to evaluate the capacity of the network to modify the amplitude and the frequency monotonically with the input.

Results. Fifteen runs were carried out until convergence (approximately 1000 generations per run) with populations of 100 randomly initialized individuals. All runs converged to networks capable of generating regular oscillations whose frequency can be modulated using the tonic drive (BS). One of them, the best, had the same topology as the network found in [14], but with small differences on the weights and thus on the neural activity.

Since the second design step is to evolve the connections between multiple neural oscillators by projecting internal connections to neighboring oscillators, it is important that the number of internal connections (from a neuron of type A, B or C to neurons of type A, B, C or M) is as small as possible. We therefore chose another one, with a smaller range of frequencies but with only 8 internal connections instead of 14. Figure 2 shows the topology of the chosen evolved network. The level of tonic drive (BS) can be varied between 0.2 and 1.13 (the frequency and the amplitude increase monotonically in this range of input), and a range of frequencies from 0.8 to 3.3 Hz can thus be covered. The amplitude then ranges from 0 to 0.21. Each neural oscillator will drive one motor in the mechanical structure by using the difference between left and right motoneurons

(M_L and M_R)): i.e. the desired angle (in radians) is defined as $\theta = \gamma(M_L - M_R) + \delta$ where δ is an angular offset and $\gamma = 5$.

2.2   Co-evolution of Body and Brain

In this section, the structure of simulated multi-unit robots and their loco-motion controllers are co-evolved. The locomotion controllers are constructed out of the neural oscillators presented in the previous section, with one os-cillator per robotic module. The coupling between the different neural oscil-lators is based on *synaptic spreading* similar to that used to model the lam-prey's CPG in [5,9,14]. The idea is to project the connection between two neu-rons within one oscillator to corresponding neurons in neighboring oscillators. The synaptic spreading can be to the nearest neighbor oscillator only or even further.

   An interesting property of this type of coupling is that it is specified by relatively few parameters compared to a scheme using all possible connections between different neural oscillators. Synaptic spreading only requires for each connection within one oscillator integers determining the extents of the projec-tions (i.e. to first, second, third, ... neighbors).

Genetic Algorithm and Encoding. The genetic algorithm is the same as the one used for the neural oscillator but with several different parameters: $P_C = 0.2; P_M = 0.05; P_P = 0.0$, and a probability of structural mutation $P_{struct} = 0.025$ is added.

   The genotype is a tree (no cycle allowed) as in [8] and [18], each node rep-resenting a module (see fig. 3 for details). It is thus a direct encoding which strongly correlates the phenotype and the genotypes. The simulation and evolu-tion environment uses the genotype as the internal representation of the robot. In addition, a chromosome of real numbers, which we will call High Level Pa-rameters (HLP), encodes the left and right input drive (BS). These genes specify the best BS in order to achieve the fastest locomotion. Crossover is simply done by exchanging two randomly chosen subtrees of the parents.

Fitness Function. The individuals are tested during 30 simulated seconds in a simulated 3D world. The simulation is a physics-based simulator (articulated rigid body dynamics with a simplified hydrodynamics model) built using ODE [19]. Throughout the evaluation period, the robot must cover as long a distance as possible. As the simple measure of the straight distance from the initial location to the final location seems not to be sufficient because of the risk to return to starting point after 30 seconds, the fitness function is based on the covered distance plus the cumulated distance:
$f = \alpha \cdot \|\boldsymbol{p(N)} - \boldsymbol{p(1)}\| + \beta \cdot \sum_{t=1}^{N-1} \|\boldsymbol{p(t+1)} - \boldsymbol{p(t)}\|$ where $p(t)$ is the tth point sampled on the trajectory of the robot, N is the total number of recorded points, and $\alpha = 1$ and $\beta = 0.3$ are coefficients that modulate the weights of the absolute and integrated distances.
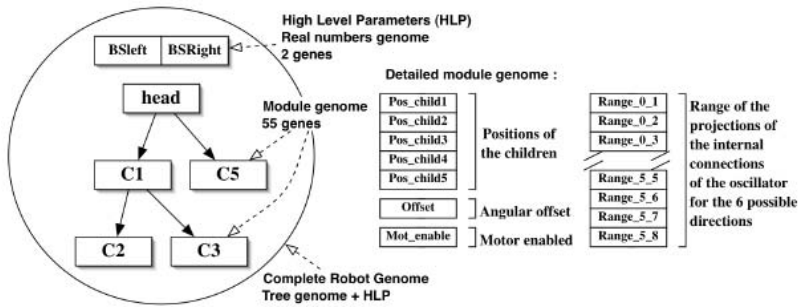
Fig. 3. Genome of a robot: It is designed as a tree with a node for each module of the robot. As we use a tree, each module has one and only one face which is actuated (the face attached to the parent) and therefore only one neural oscillator. Thus each node contains a chromosome of 55 genes containing the following information: positions of the children (5 binary genes whether or not a child module is attached to one of the 5 faces), angular offset of the joint between the module and its parent (1 real number gene), one binary gene indicating if the motor of the module must be actuated, and 48 integer genes encoding the synaptic spreading, i.e. coding the extent of the projections in the six possible directions for each oscillator.

2.3   Results

Two sets of 5 runs each, called A and B (view table 1), were carried out until convergence with a population of 100 robots. The first generations of the runs A and B start with randomly initialized populations of 5-unit robots, and 10-unit robots, respectively. Of course, the size of the robots can then be increased or decreased during the evolution. All runs converged to interesting robot structures capable of progressing in water (Table 1). The minimal number of generations required for the convergence is 128 (B5) and the average number of generation is 860.

Description of the most efficient robots. The only two robots (A5 and B3) that are identical among all runs are, interestingly, also the most efficient, the simplest and the smallest of all. As they are the only two robots that have the same shape and as they are the most efficient, they probably correspond to a good optimum in the search space. They consist of 5 modules forming two limbs of unequal sizes (fig. 4 top left). All the oscillators are activated although this can appear useless since only the joint number 2 is really generating thrust. However, all contribute to the general behavior, and the fact that the limbs oscillate on themselves reduces their rubbing, and helps the locomotion.

Diversity. Except for the best solutions of runs A5 and B3 mentioned above, all runs evolved to different types of robots. The number of parameters to be optimized and their relatively large intervals of values, make the search space very large. With the ten runs presented here, we thus explore only a small part of the search space. That, and the topology of the search space which might have many local optima, probably explains the diversity of the results.

Table 1. Table of the ten runs. *Velocity* means : Average velocity on the XY plane of the best robot in [m/min] calculated during 15 seconds after 15 seconds for stabilization. *INM* means Initial Number of Modules and specifies the size in terms of modules of the robots of the first generation. *FNM* means Final Number of Modules and is the final size in terms of modules of the evolved robots.

Set	ID	INM	FNM	Fitness Max.	Fitness Av.	Velocity
	1	5	6	4.81	4.76	9.08
	2	5	6	4.08	3.98	6.8
A	3	5	5	5.32	5.22	10.03
	4	5	6	4.8	4.71	9.42
	5	5	5	6.3	6.23	10.58
	1	10	8	4.6	4.53	9.13
	2	10	9	4.63	4.54	9.14
B	3	10	5	5.46	5.41	10.50
	4	10	9	3	2.99	6.12
	5	10	8	2.25	2.2	4.79

Evolution of the number of modules. The five runs of the set A did not add or remove large number of modules to or from robots compared to the initial populations and to a lesser extent it is also the case for runs of the set B. Simple and effective solutions are quickly found by the GA and the addition or the withdrawal of modules is almost never done. It is primarily the neural network which is optimized. This "inertia" of the size is probably explained by the fact that adding or removing a module constitutes a significant perturbation of both the dynamics of the body and of the global neural network.

Symmetry. In nature, the majority of animals have an axis of symmetry. For an efficient, controllable and rectilinear locomotion it seems to be necessary. However, we made the choice not to force this symmetry and to see if it appeared spontaneously. That was not the case here maybe because for having and keeping symmetrical structures two mutations must appear at the same time and symmetrically which is unlikely (see [21] for a related study). To go straight in spite of this handicap, all robots follow spiral trajectories turning on themselves. As the center of this spiral is a line, the robots go as straight as possible with respect to the fitness function.

Evolution of the BS. The values of the left and right BS are evolved with the robots and can be varied between 0.2 and 1.13 (cf the neural oscillator). Half of the robots converge to the maximum value for the two inputs. Indeed, to swim as fast as possible it seems to be important to have high frequencies and amplitudes of the movements. The others robots have different values for the left and right BS.

In vertebrate animals, it is known that increasing the tonic drive from the brainstem increases the speed of locomotion. We tested if the robots which we evolved reacted in same the manner. We noted that speed indeed increased or decreased according to BS (cf Fig 4 top right) as awaited, although that must
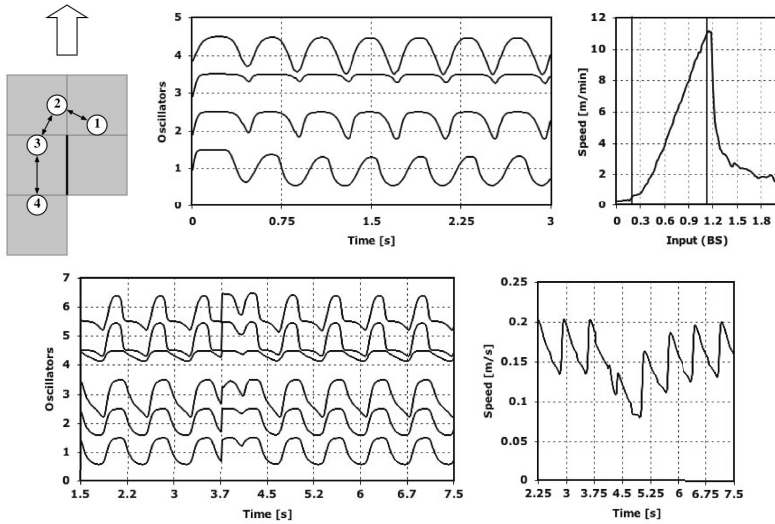
Fig. 4. Top left: Robot evolved by the run A5 (same shape as B3). **Top middle**: Neural activity of the robot A5. Each curve corresponds to the difference between left and right motoneuron of one oscillator. The four oscillators are synchronized and appear to be able to generate a lot of different signals. **Top right**: Influence of the tonic input (BS) on the velocity of the robot A5. The vertical lines determine the region in which the velocity increases monotonically with the BS. **Bottom left**: Neural activity of the robot B2. Each curve corresponds to the difference between left and right motoneuron of one oscillator. At 3.75 seconds, a random perturbation is applied to membrane potentials. **Bottom right**: Evolution of the speed of the center of mass of the robot. The oscillations are due to the periodic flutters of the limbs.

be moderated because the trajectory also changes, i.e. if the robot goes straight with maximum and symmetrical input values, it happens that the robot does not swim straight anymore when we decrease symmetrically.

Resistance to perturbations. We tested the robustness of our robots to perturbations by perturbing the membrane potentials of all neurons and setting them to random values. The neural activity and the speed of the robot B2 is shown on figure 4 on the bottom. The neural activity rapidly recovers from the perturbation and returns to steady state after 2 oscillations. The velocity also recovers although it takes a little bit more because the robot has to struggle against the water inertia. This robustness against perturbations is one of the interesting features of using CPG models for locomotion.

3   Conclusion

This article presented first results of a project in underwater modular robotics, called Neubots. Body structures and neural controllers based on central pattern generators were co-evolved for efficient underwater locomotion. The main results

are the design of neural oscillators linked together by synaptic spreading capable of producing robust signals for locomotion. The controllers can adjust the speed of locomotion by the modulation of simple signals and quickly recover from random perturbations in the neural activity.

References

1. D. Floreano, A. J. Ijspeert, T. Bertolote, and C. Mattiussi. Neubots: The robotics legacy of John von Neumann. Technical report, submitted for publication, Laboratory of Intelligent Systems, EPFL, 2005.
2. T. Bertolote and V. Hentsch, Design and prototyping of an underwater modular robot, Unpublished Master Thesis, Laboratory of Intelligent Systems, EPFL, 2004.
3. J. vonNeumann. *Theory of Self-reproducing Automata*. University of Illinois Press, Urbana, IL, 1966. Edited and completed by A. W. Burks.
4. W. McCulloch and W. Pitts. A Logical Calculus of the Ideas Immanent in Nervous Activity. *Bulletin of Mathematical Biophysics*, 5:115–133, 1943.
5. Williams, T. Phase coupling by synaptic spread in chains of coupled neuronal oscillators. *Science*, 258, 662-665, 1992
6. S. Murata, E. Yoshida, A. Kamimura, H. Kurokawa, K. Tomita, and S. Kokaji. M-tran: Self-reconfigurable modular robotic system. *IEEE/ASME Transactions on Mechatronics*, 7:431–441, 2002.
7. G. Konidaris, T. Taylor, and J. Hallam. Hydrogen: Automatically generating self-assembly code for hydron units. In *Proceedings of the Seventh International Symposium on Distributed Autonomous Robotic Systems (DARS04)*, Berlin, 2004.
8. Sims, K.: Evolving 3D Morphology and Behavior by competition. Artificial Life IV Proceedings, ed. by R. Brooks & P. Maes, MIT Press, pp 28-39, 1994.
9. Ekeberg, Ö.: A combined neuronal and mechanical model of fish swimming. *Biological Cybernetics*, 69, 363-374, 1993.
10. Grillner, S.: Neurobiological bases of rhytmic motor acts in vertebrates. *Science*, New Series, Vol. 228, No. 4696, pp 143-149, Apr. 12 1985.
11. Kamimura, A., Kurokawa, H., Yoshida, E., Tomita, K., Murata, S., Kokaji, S.: Automatic Locomotion Pattern Generation for Modular Robots. *Proceedings of the 2003 IEEE International Conference on Robotics & Automation, 2003*
12. Yoshida, E., Murata, S., Kamimura, A., Tomitd, K., Kurokawa, H., Kokaji, S.: Evolutionary Synthesis of Dynamic Motion and Reconfiguration Process for a Modular Robot M-TRAN. *IEEE International Symposium on Computational Intelligence in Robotics and Automation*, 2003.
13. Fukuoka, Y., Kimura, H., Cohen, A. Adaptive dynamic walking of a quadruped robot on irregular terrain based on biological concepts. *The International Journal of Robotics Research*, 3-4, pp 187-202, 2003
14. Ijspeert, A. J.: A connectionist central pattern generator for the aquatic and terrestrial gaits of a simulated salamander. *Biol. Cybern.*, Vol. 84:5, pp 331-348, 2001.

15. M. Komosinski, S. Ulatowski, Framestics: Towards a simulation of a nature-like world, creatures and evolution. *ECAL 1999*, pp. 261-265, 1999
16. H. Lipson, J.B. Pollack, Automatic design and manufacture of robotic lifeforms, in *Nature*, 406:974-978, 2000.
17. J.C. Bongard, R. Pfeifer, Repeated structure and dissociation of genotypic and phenotypic complexity in artificial ontongey, in *Genetic and Evolutionary Computation Conference*, p. 829-836, 2001.
18. Marbach, D., Ijspeert, A. J.: Co-evolution of Configuration and Control for Homogenous Modular Robots. *Proceedings of the Eighth Conference on Intelligent Autonomous Systems (IAS8)*, pages 712-719. IOS Press, 2004.
19. Smith, R.: Open Dynamics Engine (ODE), webpage: http://www.ode.org/.
20. GAlib, webpage: http://lancet.mit.edu/ga/
21. Bongard, J.C. and C. Paul, Investigating Morphological Symmetry and Locomotive Efficiency using Virtual Embodied Evolution, in *From Animals to Animats: The Sixth Int. Conference on the Simulation of Adaptive Behaviour*, pp. 420-429, 2000.