In Proceedings of the 9th ESA Workshop on Advanced Space Technologies for Robotics and Automation
'ASTRA 2006' ESTEC, Noordwijk, The Netherlands, November 28-30, 2006

Objects recognition through motion comparison and environmental interaction for autonomous robots: an Evolutionary Robotics approach

Federico Vicentini, Alberto Rovetta
Robotics Lab, Mechanics Dept., Politecnico di Milano
P.zza Leonardo da Vinci 32, 20133 Milan (Italy)
{federico.vicentini,alberto.rovetta}@polimi.it

Abstract.
In unknown environments, context learning and adaptive decision making are desired generic skills for an autonomous agents devoted to unsupervised explorations. Regardless the peculiar kind of the environment to explore, an autonomous agent is likely to be equipped with a basic set of sensory apparatus and plays a number of actions involving some selective interaction over the searched environment after a detective (cognitive) phase. This general process may be performed by an autonomous robot suitably developed for a predefined set of skills that should let the robot adapt to any unexpected situation. Hence, the development of the desired capabilities may go through a representation of both the environment and the actions to perform. An Evolutionary Robotics framework is presented, setting an experiment fit for recognition of non-inert, possibly active, objects. The task involves environment exploration and classification in a context of self-perception. In this scenario, the robot is able to perform comparison activities for understanding the mutual differences in locomotion patterns between object motion and its own movements. These capabilities are due to the structure of a neural controller, able to recurrently process the information about motion and environment perception. The motion comparison is used for classification purpose before planning any activation behavior.

INTRODUCTION.

Environmental exploration is one of the capabilities most largely issued in autonomous robotics, mainly due to possibility to provide a mobile robot with large sensor equipment, effective locomotion features and adaptive strategy for overcoming environmental constraints (see, for instance [1],[2]). As an exhaustive example of listed features, the mobile robots for planet exploration granted the scientific community with huge enhancement in ground sampling, image collection, experimental trials with carried payloads. However, the robots used so far have been implemented with a limited degree of autonomy in exploration and environment recognition. Most of the mobile robots applications involve somehow online feedback by external driving for detection activity.
Therefore, it may be very interesting to consider an explorative context for a largely autonomous agent, able to search for interesting conditions in the explored environment and

to take adaptive decisions. Some of decisional criteria are likely to be defined by design, but many others may be unpredicted and eligible for agent independent elaboration and statement. In this way, any adaptive behavior - developed in an unforeseen combination of eventually already experienced elementary events - yields a significant improvement for robotics in intelligent control state-of-the-art. The basic features that may be issued and experimented for getting a suitable adaptive behavior are the capabilities of classification and modification of the environment. Any autonomous agent able to self-determine the status of the environment and to chose whether and what interaction to produce on it, should react to sensorial inputs with full awareness of any detected event. To this end, the design process should provide the agent controller with all the perceptive and cognitive skills for all the expected situations. However, the context may be unknown and the variability of any possible interaction between the environment and the active agent may be far more extended than any predictable definition. Thus, the technique used for developing the agent intelligence should effectively tackle with unpredictability whenever no preloaded behavior rules can be coded. In this paper we propose a self-defined emergent behavior approach through which the agent evolves a strategy for accomplishing the issued problem without any forced solution. The desired behavior is in fact defined by an high level principle that gives reason of both the requirements and the constraints but not of the operative strategies. An evolutionary selective process is therefore used to empirically explore a large number of strategies. A number of fit controllers for the agents are selected and iteratively improved generation after generation. In this way the development framework provides the agent with all the environmental and procedural information useful for getting an autonomously determined solution for the problem, respecting the desired general behavior and the requirements. The technique needs of a simulated environment where the agent is modelled in kinematics, dynamics, sensory apparatus and functional subsystems, like signal emission. In addition, the environment is represented with a degree of accuracy fit for the controller evolution. Since the interaction mechanisms are empirically tested during the simulation, the accuracy of the description is directly related to the way the agent perceives (and acts on) the environment. Moreover, the unpredictability involves a certain degree of uncertainty also in environment modelling. Since an evolutionary process gains robustness in optimizing the required strategy in presence of widely varying model parameters, the model should not use excessive computational resources. The environment and task description are provided in session TASK. The system description with the controller architecture and the evolutionary process are discussed in sessions CONTROLLER and FRAMEWORK respectively. Experimental results are finally presented in session RESULTS.

THE TASK.

The target of the experiment is obviously the set up of a fit controller able to perform the detection of some objects in an unknown environment and the activation of a subset of them. In this frame, “activation” has the meaning of an action by the agent on the object qualitatively different from any other during the exploration. Since the task should be representative of the exploration feature (short term reactive skill), the activation feature (long term memory skill) as well as of the classification feature (cognitive and comparison skill), the simulation provides the basic elements necessary to reproduce and test the desired skills. Hence, the agent and the objects populating the environment are likely to represent only the minimal interaction mechanisms related to the given scenario of search and activation. First, the agent is able to explore the environment by moving according to the evolved strategy. Then, the sensory equipment is limited to light sensors, providing distance and light intensity of any light emitting object inside the range of perception of the sensors. Finally, the agent activation capabilities are limited to sound emission (and linked sound feedback with sound sensor on board). Given the basic agent skills, the objects representing the environment are designed to address corresponding interaction mechanisms and, at the same time, to cover the range of possible expected typology of objects in an unpredictable environment. The objects are set as lights moving according to a motion pattern distinctive of the nature of the represented object. All the light-emitting objects are similar in shape and emission before eventually getting in contact with the agent and being activated. Once activated by sound, the light-emitting objects may change their color.

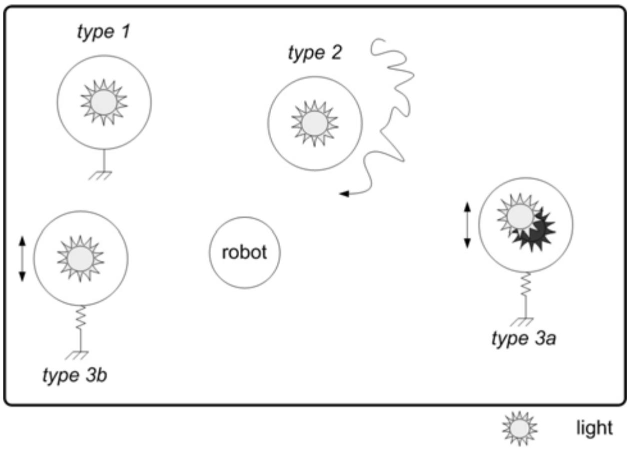

Figure 1: Ideal environment representation. The 4 types of objects are represented as lights surrounded by their range of emission (distance at which the agent starts to perceive the light). Type 1 object has fixed joint, type 2 moves on a random trajectory, type 3 and 4 are connected with an ideal non dissipative spring. The black thick lines depict the graphical boundary of the environment. The frame is however unlimited and toroidal.

The resulting experiment is therefore conceptually similar to the desired requirements and not directly related to any particular hardware or equipment. Since many sensors may perceive and process a wide range of objects and situations, the experimental run is set up in an universal high level representation. Any real world experiment can be drawn from this conceptual one, modifying the kind of input/output equipment. The selected simulation set up is related to the *s-bot* platform and runs in a largely tested simulator [3] reliably ported on the real platform. So the hardware aspects (sensors calibration, signal models test and validation, etc) are suitably established.

The cognitive classification skill is developed through motion comparison. Since the agent's light sensors provide information about relative position and intensity of incoming signal, one of the simplest way for analyze and classify objects is to compare the object motion with the agent own movement. The controller has a direct feedback of the motion produced by the activation of the steering wheels. The controller is updated at each simulation step computing the new mutual position. In this way the controller has the chance to evolve a strategy for processing a differential information during the simulation about the object displacement and its own one. The typology of the objects is now based only on motion pattern, so this discrimination allows the environment to be populated with 4 kinds of potentially (or not) interesting objects.

The object typologies are :

1. motionless objects: they are passive part of the environment. They attract the agent but they are also inert. The agent should learn to ignore them.
2. randomly moving objects: they are active part of the environment. They attract the agent but they also are inert like type 1. The agent should learn to ignore them.
3. periodically moving objects: they are active and probably interesting objects. They attract the agent and they behave differently from the other objects. In some way they interact with the agent changing the color of their light if the agent produces sound when it is nearby:
    a. changing light objects: "living" objects. The agent should learn to stay around them after the detection and activation. They are the most relevant target of the exploration.
    b. unchanging light objects: "dead" object. Despite their interesting motion behavior they are not relevant for the exploration.

One way to find the exploration target (3a) is to travel around the environment, selecting the potentially interesting objects (3x), then to actively interact with them to induce a shift in their behavior (light color), if there is one.

The type 2 object moves along a random trajectory inside a limited area. Type 3 objects amplitude and centre of oscillation position prevent any overlapping of emission ranges. The oscillation is one axis directed. The incremental position for type 2 objects, and oscillation parameters for type 3 objects are set up considering the maximum speed of the agent and the chance to compare objects motion without losing contact because of extra displacements.

Any type of object is reproduced in each trial with initial random positioning in order to prevent any association between the type of object (target of the classification skill) and the position (accidental correspondence). The agent starting orientation is also randomly initialized in order to average the effect of the direction of exploration during the first simulation steps.
Finally, each light object initially emits yellow light and switches to red light only if belonging to type 3a and being activated by sound after a classification activity by the agent. There is no chance to initially distinguish type 3a from 3b unless to activate the objects.

THE CONTROLLER AND THE SIMULATED AGENT.

The controllers are evolved in a simulation environment which models some of the hardware characteristics of the real *s-bot*. The *s-bost* (see Fig. 2a) are small wheeled cylindrical robots, 5.8 cm of radius whose mobility is ensured by a differential drive system (see [4] for details). The robot uses the ambient light sensors (AL1) and (AL2) positioned at ±67.5° with respect to the orientation of the robot. Light sensor values are simulated through a sampling technique (see [5]). Besides the intensity with respect to the direction of incoming light, AL1 and AL2 are able to perceive the color of the light. The color is mapped into $\zeta_i \in \{0.0;1.0\}$ in order to estimate two different colours.

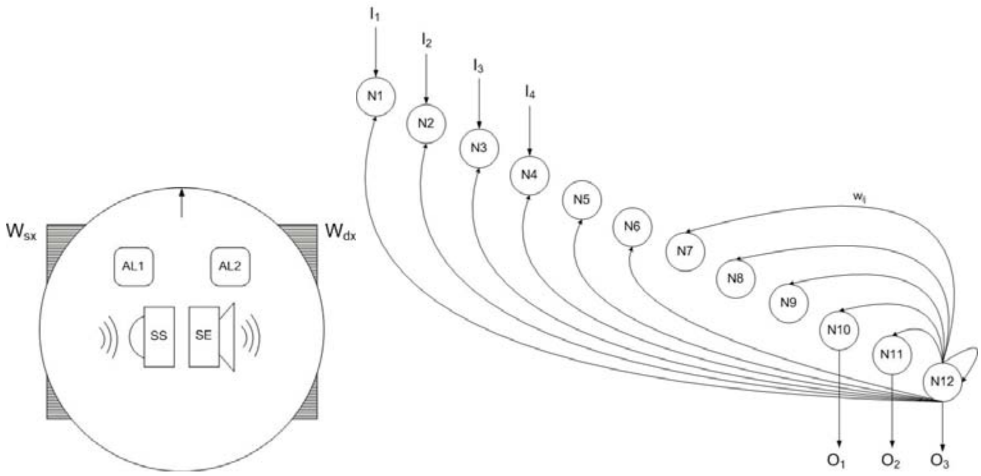

Figure 2: on the left, the *s-bot* equipment schematic representation. On the right, the fully recurrent neural network used as a controller. Inputs I and outputs O are connected to extreme neurons, the synapses and complete connections are depicted only for neuron N12. Every neurons has the same synapses structure.

The robot is also equipped with a loud-speaker (SE) and one omni-directional microphones (SS) that are situated in the centre of the body. Sound is modelled as an instantaneous, additive field of single frequency with two level intensity $\eta_i \in \{0.0;1.0\}$. Sound intensity is regulated by the firing rate of neuron N12 (see Fig. 2b). Concerning the function that updates the position of the robots within the environment, we employed the Differential Drive Kinematics equations, as presented in [6]. A 10% uniform noise is added to all sensor readings, the motor outputs and the position of the robot.
The agent controller is made up of a fully recurrent network of twelve neurons. The sensory neurons receive input from the agent's sensors equipment: N1 from SS, N2 and N3 for light intensity from AL1 and AL2, N4 for light color. Each output neuron (N10, N11 and N12) receives incoming synapse from every other neuron. There are in this way direct connections between sensory and output neurons so that the controller can better perform the self-perception capabilities. The network neurons are ruled by the following state equation:

$$\tau_i \dot{x}_i = \begin{cases} -x_i + \sum_{i=1}^{N} w_{ij}\sigma(x_j) + g_i I_i & i \in [1,4] \\ -x_i + \sum_{i=1}^{N} w_{ij}\sigma(x_j) & i \in [5,12] \end{cases} \quad \text{with } \sigma(x_j) = \frac{1}{1+e^{-g_j(x_j+\beta_j)}} \tag{1}$$

where, using terms derived from an analogy with real neurons, τ_i is the decay constant, x_i represents the cell potential, I_i is the intensity of the sensory perturbation on sensory neuron *i* amplified by a gain g_i, w_{ij} the strength of the synaptic connection from neuron *j* to neuron *i*, β_j is the bias term, and $\sigma(x_j)$ represents the firing rate. The cell potentials x_i of neurons N10 and N11, mapped into [0,1] by a sigmoid function σ and then linearly scaled into $[-6.5,6.5]$, set the robot motors output. A simple generational genetic algorithm is employed to set the parameters of the networks [7]. The population contains 50 genotypes. Each genotype is a vector of 169 real values (i.e., 144 connection weights, 12 decay constants, 12 bias terms, and 1 gain factor). The following parameters are genetically encoded (see next session): (i) the strength of synaptic connections w_{ij}; (ii) the decay constant τ_i of each neuron; (iii) the bias term β_j. The decay constant τ_i of the sensory neurons and of the output neurons are non-synchronous with the time constant of system update on the real *s-bot* processor. This feature allows the controller to process the signals and potentials with different dynamics than the robot update in motion. Cell potentials are set to 0 any time the network is initialised or reset, and circuits are integrated using the forward Euler method with an integration step-size of *dt*=0.1s.

THE DEVELOPMENT FRAMEWORK: GENETIC ALGORITHM AND DESIGNED FITNESS FUNCTION.

The requested behavior is coded into a principle function (fitness function) through which the designer sets the desired high level actions but without specifying any predefined rule. The fitness function provides only a reward/punishment policy for computing a score for the agent behavior during the simulation. The fitness score provides the evaluation value for the genetic optimization process. Each of the 50 individuals (corresponding to an empirically created controller) is tested during the simulation for 100 trials. The resulting score f_e gained in each evaluation is averaged on the 100 trials in order to purge any casual effect that may reward a sub-optimal individual. In each of the 2000 generations the individuals are tested and the best performing individuals breed the population of the following generation according to genetic operators. The genome coding the controller parameters is modified with recombination and mutation operators. In this way the controller skills are transferred to the child generation and the population continues the exploration of fitness function domain towards the optimal solution. At last generation, the best scoring genome is assumed to be the best performing controller and is re-evaluated to check the reliability of the obtained solution.
In the current experiment the agent increases its score according to the quality and the cognitive complexity of its actions. A relatively low score is paid to the exploration skill since it is supposed to be the most time-consuming activity. An higher score is gained when the agent enters in perception range of the light emitted by the objects, and so starts to classify. The highest score is given to the activation (by sound) of the target object (3a) in the environment.
Hence, the task accomplishment is related only to the agent status during the simulation and the corresponding score. The agent does not receive any information about its status and the quality of the action it is currently performing. The optimization engine records the behavior through the fitness score and no online feedback is provided to the controller. In this way the agent is free to act according to its own strategy. Only the most effective one has the chance to develop and to be refined along the generations through the breeding mechanism. Since the agent does not check its actions, the designer is free to set any rewarding mechanism. In this experiment the status of the agent is determined by checking the motion and activation conditions. The fitness function is so figured out of a rule tree according to desired behavior in each of the conditions considered. If the agent behaves properly it is rewarded, otherwise it does not gain any score. Decreasing scores for punishment are not effectively fit for this kind of experiment especially during the first generation, when all the individuals are randomly seeded and the corresponding controllers are lacking in any consistent strategy. This may cause a failure in each individual in gaining a minimal or negative score, preventing the population to move towards a possible optimal solution. This is also known as bootstrap problem (see [8],[9]).
The fitness function reward policy is shown in Fig. 3.

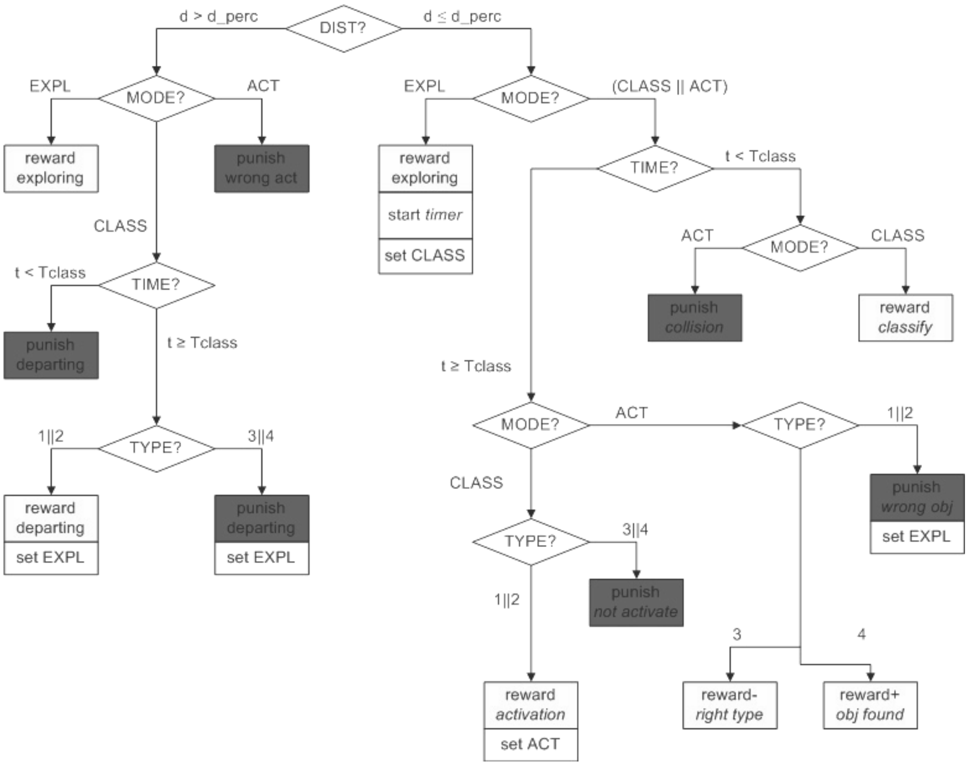

Figure 3: a representation of the rewarding mechanism for the fitness score. Each component of f_e is computed by checking the status of the agent, according to topological or chronological conditions. Recall that the agent does not get any cue of the status where it is placed in this classification. For final score assignment, the criterion is based on the type of the object eventually activated. Again the agent has not any information on the type of the object.

At each time step the agent scores the contribution to fitness $f_e(t) = K_{expl} f_{expl} + K_{class} f_{class} + K_{act} f_{act}$, where only one term at a time - related to the current status - is non null.

RESULTS.

Ten evolutionary runs were launched for 3000 generations each with different initialization seeding. The best evolved genotype, i.e. the genome of the last generation, for each run was

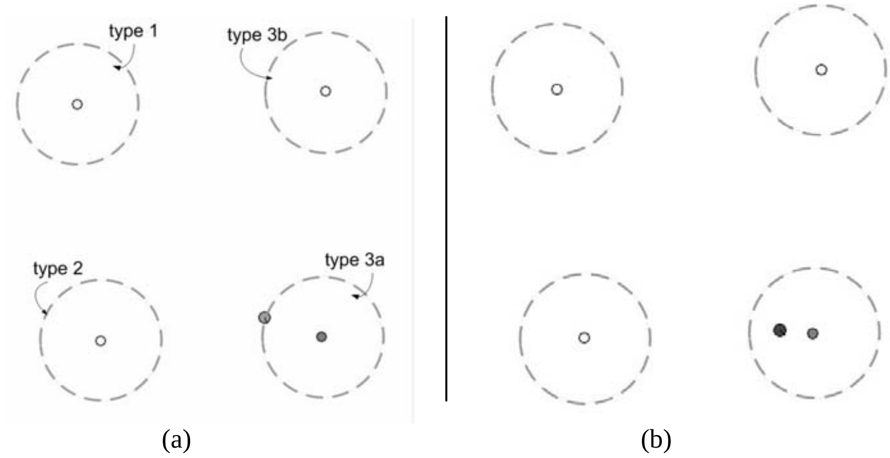

Figure 4: the accomplishment of the task for the best evolved genotype, shown in 2 consecutive steps. (Object type in this particular trial is shown for convenience.) From the left, (a) the agent enters the range of perception of light, the object in the bottom right corner is of type 3a and it switches to red color (darker in figure). (b) The agent stays near the changed object, emits sound (represented by the darkest color of the agent) and rotate around without moving away.

re-evaluated in order to check whether the over-estimation of the fitness score may affect the reliability of the accomplishment of the task.
The best absolute behaving genotype is afterwards coded for further 100 post-evaluation trials. The measures for selecting the best overall genome and the post-evaluation statistics are not shown here, and they are beyond the purpose of the present discussion. The evaluation of the performance for the best behaving genotype allows to consider the achievement of the first basic capabilities in exploring and classifying the environment. Through a simple graphical interface, we can observe that the task is accomplished with a fast moving phase of exploration. Then the agent, when it enters the light emission range, starts to detect the type of the object, and whenever the object is of type 4, the robot switches the behavior to a sound emitting and spinning movement in the same place (see Fig. 4). As a result, the recognition of the object after the activation and the change of the color by object of type 4 is very robust and the corresponding behavior very different from the explorative one. The same controller can therefore apply different skills during its interaction with the environment strongly connected to the object of interaction. We consider in this experiment that the behavior is not affected by casualty and the capabilities of the agent are reliably achieved by the empirical evolutionary development.

CONCLUSIONS.

In this paper we presented a conceptual framework for developing complex capabilities in autonomous agents. The complexity of the skills is mainly due to the possibility of dynamically change the behavior during the working activity in an unexpected environment. Many tools and techniques are suitably used to gain this target, but here we focused on the Evolutionary Robotics in order to investigate the effectiveness of letting the agent to develop its own skills in an empirical way inside a minimal simulated environment. In this way, we built a model of the basic interactions between the agent and the environment, and represented this latter by the very few features useful in spanning the classes of objects that the agent is expected to encounter during a exploration activity. The related experiment was therefore set up in order to couple the minimal-conceptual description of the environment with the most reduced modelling of the agent equipment. Although the poor set up, the neural controller evolved in the experiment shows a very large and structured range of capabilities, given the fact that the same controller perform sub-task very different one from another. We hope this will help the autonomous robotics practitioners in considering empirical tools like an effective technique for enlarging the capabilities of a largely autonomous agent.

REFERENCES.

[1] Muscato G., Caltabiano D., Guccione S., Longo D.,, Coltelli M.,, Cristaldi A.,, Pecora E.,, Sacco V.,, Sim P.,, Virk G.S.,, Briole P.,, Semerano A., White T. “ROBOVOLC: a robot for volcano exploration result of first test campaign” *Industrial Robot: An International Journal* 2003, 30 (3) P. 231-242

[2] Blynel J., Floreano D. “Exploring the T-Maze: Evolving Learning-like Robot Behaviors CTRNNs” *Proceedings of EvoROB2003 2nd European Workshop on Evolutionary Robotics*

[3] Vicentini F., Tuci E. “Swarmod: a 2d s-bot's simulator” *Technical Report TR/IRIDIA/2006-005*, IRIDIA, Universitè Libre de Bruxelles (2006) ISSN 1781-3794

[4] Mondada F., Pettinaro G. C., Guignard A., Kwee I. W., Floreano D., Deneubourg J., Nolfi S., Gambardella L.M., Dorigo M. “Swarm-Bot: A New Distributed Robotic Concept” *Autonomous Robots 17, 193–221, 2004 Special Issue On Swarm Intelligence*

[5] Miglino O., Lund H., and Nolfi S. “Evolving mobile robots in simulated and real environments” *Artificial Life*, 2(4):417–434. (1995).

[6] Dudek G. and Jenkin M. “Computational Principles of Mobile Robotics”. Cambridge University Press, Cambridge, UK. (2000).

[7] Goldberg, D. E. “Genetic Algorithms in Search, Optimization and Machine Learning”. Addison-Wesley, Reading, MA. (1989).

[8] Floreano D. and Urzelai J. “Evolutionary Robotics: The Next Generation”. *In T. Gomi (ed.), (2000) Evolutionary Robotics III, Ontario (Canada): AAAI Books*

[9] Nolfi S. and Floreano D. “Evolutionary Robotics: The Biology, Intelligence, and Technology of Self-Organizing Machines”. MIT Press, Cambridge, MA. (2000).