ESANN 2011 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 27-29 April 2011, i6doc.com publ., ISBN 978-2-87419-044-5. Available from http://www.i6doc.com/en/livre/?GCOI=28001100817300.

Thresholds tuning of a neuro-symbolic net controlling a behavior-based robotic system

M. Staffa[1] , S. Rossi[2], M. De Gregorio[3], E. Burattini[2]*

1- Dipartimento di Informatica e Sistemistica - Università di Napoli “Federico II”
Piazzale Tecchio, Naples – Italy
2-Dipartimento di Scienze Fisiche - Università di Napoli “Federico II”
Via Cintia, Naples – Italy
3- Istituto di Cibernetica “E. Caianiello” – CNR, via Campi Flegrei, Pozzuoli - Italy

Abstract. In this paper we present the results obtained by adopting an evolutionary approach to tune some critical neuron thresholds of a neuro-symbolic net that regulates the overall emergent behavior of a behavior-based robotic system.

1 Introduction

Due to the variety and complexity of real environments autonomous robots need control systems to assure an efficient use of their limited sensorial and cognitive resources and to balance sensors elaboration and actions execution. In order to deal with these problems and inspired by human attention mechanisms, in [1] we proposed an architecture, named Adaptive Innate Releasing Mechanism (AIRM), for Behavior Based Robots (BBR) capable of adapting the frequency of the sensors sampling rate both to the dynamic environment and to the internal states.

This kind of adaptation may be interpreted as a selective attention mechanism that filters information in order to focus on salient events. Some results, providing an improvement in the BBR performances, compared with the same architecture without AIRM, are reported in [2]. Moreover, in order to realize a controller able to manage in real time the reading rate adaptivity, we introduced a Neuro-Symbolic Net (AIRM-net) [3] that implements this kind of attentive controller.

In this paper, we adopt an evolutionary algorithm, called Differential Evolution (DE) [4], to find, in the space of possible solutions, the best setting of some critical parameters of the net (thresholds), regulating the overall emergent behavior. We show how this kind of algorithm is able to find the thresholds values producing the best fitness and maintaining the implicit constraints introduced by the AIRM-net.

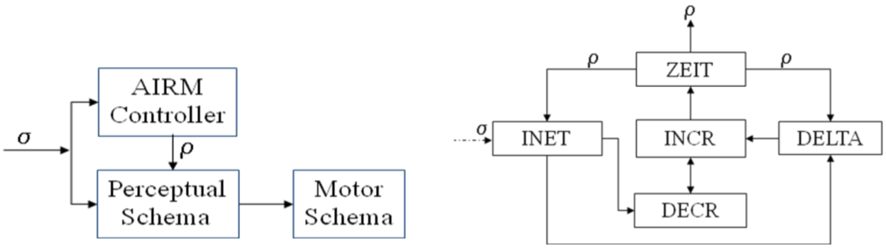
2 The AIRM-net approach

A BBR is usually characterized by a set of behaviors assembled to accomplish the desired activity. In the Schema Theory approach [5], a behavior is represented by a Perceptive Schema (PS), a Motor Schema (MS) and, in some cases, it can be controlled by a releasing mechanism triggering its activation. Our model adds to this

* The research was partly supported by the European Community’s Seventh Framework Programme (FP7/2007-2013) under grant agreement no.216239.

ESANN 2011 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 27-29 April 2011, i6doc.com publ., ISBN 978-2-87419-044-5. Available from http://www.i6doc.com/en/livre/?GCOI=28001100817300.

behavior schema a clock mechanism (AIRM), which samples data coming from sensors (Fig. 1(a)) with an adaptive frequency depending on the sensor input changing rate. This mechanism drastically reduces the computational load associated to the sensors elaboration. In order to face real time applications, in [3] we implemented the clock mechanism by means of a Neuro-Symbolic net (AIRM-net) (Fig. 1(b)). This net can be automatically designed by the Neuro-Symbolic Behavior Modelling Language (NSBL) [6] that allows to express propositional logical inference and to translate them into the logically equivalent neural network.



(a) (b)

Fig. 1: (a) Behavior schema; (b) AIRM-net controller schema.

The AIRM-net is characterized by a time interval, named *clock period* or p_β, used to space out two successive sensors readings. p_β is generated by the ZEIT module (Fig. 1(b)) and it is initially set to a maximum value (p_{bmax}). The ZEIT module interacts with the INCR and DECR modules in order to change the clock period according to the increasing or decreasing input variations coming from INET module. Furthermore, by means of a releasing function $\rho(p_\beta)$, the ZEIT module communicates when the behavior has to process sensory inputs.

2.1 AIRM-net activity

The AIRM-net modules are sketched in Fig. 2. INCR and DECR modules are controlled by INET and DELTA modules. INET conveys the input signal σ, read by the sensor at time t and t-p_β, to the DELTA and DECR modules. The DELTA module is activated when the sensor signal increases between two successive readings. The rate variation can be evaluated with respect to the salience $\Delta\sigma_t/\sigma_t$ (where $\Delta\sigma_t=\sigma_t-\sigma_{t-p\beta}$) or to the temporal incremental ratio $\Delta\sigma_t/p_\beta$. The INET module activates the DECR module if the input signal decreases. The interaction between DECR and INCR modules and then with the ZEIT module provides respectively an increasing or decreasing of the clock period.

In this paper we mainly deal with the INCR module (see Fig. 2) that provides a sort of focus of attention mechanism by reducing the period p_β. This module is formed by two layers of `type_N` neurons (*iCL* and $in_{i,j}$) characterized by the following transfer function:

$$type\_N_i(t) = 1\left[\sum_{j=1}^{K_i} a_{i,j} * j(t-1) - th_i\right]$$

where neuron j is either a `type_N` neuron or a `type_Δ` neuron.

ESANN 2011 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 27-29 April 2011, i6doc.com publ., ISBN 978-2-87419-044-5. Available from http://www.i6doc.com/en/livre/?GCOI=28001100817300.

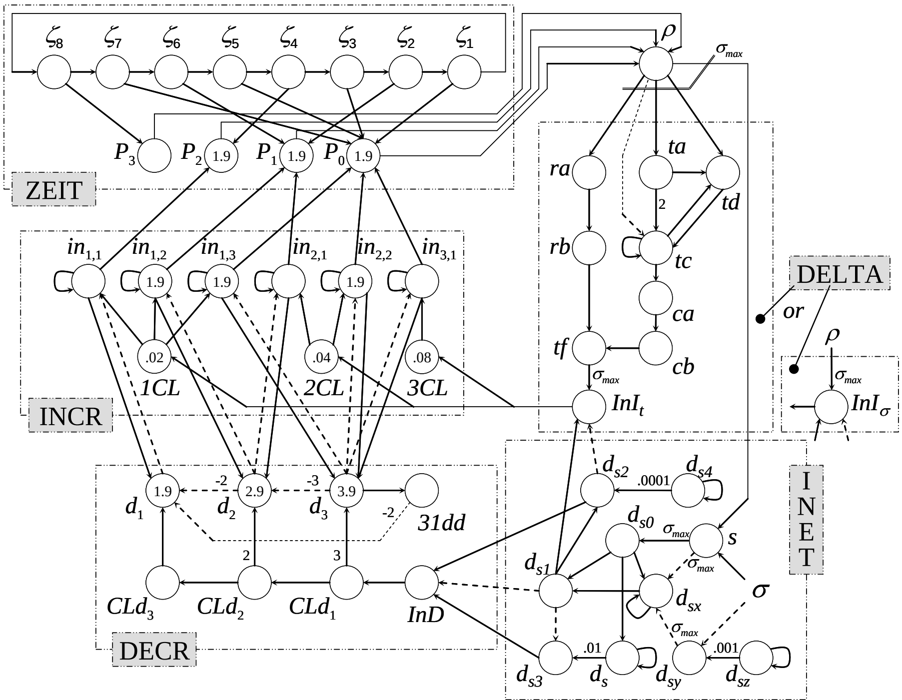


Fig. 2: the AIRM-net

A `type_Δ` neuron, such as InI_x neurons, evaluates the rate variation ($InI_\sigma=\Delta\sigma_t/\sigma_t$ or $InI_t=\Delta\sigma_t/p_\beta$) and fires on the *iCL* neurons of the INCR module. Without loss of generality, we chose an updating policy for the clock that decreases its period according to the powers of two. Hence, we need $n=\log_2(p_{bmax})$ *iCL* neurons. If *iCL* neuron fires, the new period will be equal to $p_{bmax}/2*i$. In a first implementation of the net we experimentally determined the values for the *iCL* thresholds, depending on several factors such as the sensors precision, the special features of the environment and the behavior goal. The only constraint was that the *iCL* thresholds had to be in ascendant order in a way that the decreasing process be gradual and proportional to the variation of the input signal.

3 Thresholds tuning

In order to get a good performance for our robot and to extend and generalize the AIRM-net, the *iCL* neuron thresholds, regulating the period adaptation process, are tuned through an evolutionary approach; in particular, we deploy the DE algorithm. This algorithm gradually achieves the robot control system as an optimization problem. At each generation it produces a new population of candidate solutions combining the existing ones according to a mutation operator *F* (i.e. maintaining the most suitable solution for the optimization problem). Then, in order to increase the diversity of the perturbed parameter vectors (individuals of the new population), a crossover factor *CR* is introduced [4]. The general formulation of the problem is to

ESANN 2011 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 27-29 April 2011, i6doc.com publ., ISBN 978-2-87419-044-5. Available from http://www.i6doc.com/en/livre/?GCOI=28001100817300.

consider a *fitness* function that evaluates the system performances depending on the choice of the critical values of the *iCL* thresholds, controlling the AIRM-net, and to solve the minimization problem by finding the *iCL* thresholds combination that produces the best (minimum) value for the fitness. The fitness function evaluates the robot global behavior by considering some application-dependent performance measures (i.e. time to accomplish the goals, number of dangerous situations, etc.) during the interaction between the robot and the environment.

3.1 Case study

We tested our approach using a simulated Pionee-3DX mobile robot, endowed with a blob camera and sonars, and controlled by the Player/Stage tool [7]. The robot, without any *a priori* knowledge, has the task of finding food (gray circle in Fig. 3) in the environment, avoiding obstacles (black squares) and coming back to its nest, i.e., its starting point (striped rectangle).

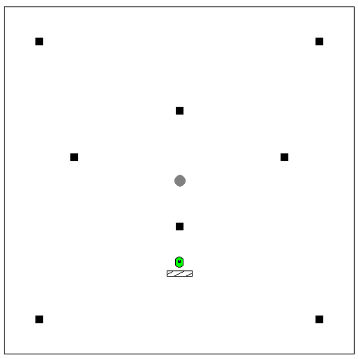


Fig. 3: Simulated environments.

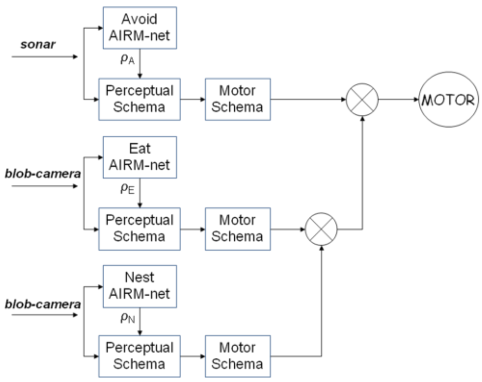


Fig. 4: BBR architecture.

This domain is a good testbed from a behavioral point of view, since it combines attractive and repulsive behaviors. The behaviors are represented by suitable AIRM-net provided respectively by InI_σ or InI_t neuron. The architecture (Fig. 4) is characterized by three behaviors endowed by an AIRM, whose outputs are combined through the classic subsumption mechanism [8]. The DE algorithm is implemented considering as individual of a population a single robot whose AVOID AIRM-net is characterized by a particular combination of the *iCL* threshold values. The DE evaluates the performance of such a robot, while changing the *iCL* thresholds, by means of the following fitness function:

$$fitn(x) = M_1 \times \frac{avoid\_count}{cc} + M_2 \times \frac{num\_crash}{cc} + M_3 \times \frac{time}{task\_time} + \\ + M_4 \times (1 - food\_count) + M_5 \times (1 - nest\_reached).$$

x is an individual of the population; *cc* is the number of executed computational cycles; *avoid_count* represents the number of calls to the AVOID behavior; *task_time* is the maximum time allowed to accomplish the task; *time* is the effective time spent

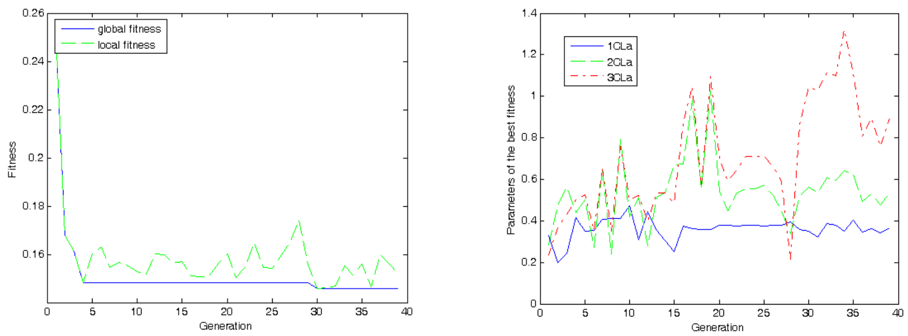
ESANN 2011 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 27-29 April 2011, i6doc.com publ., ISBN 978-2-87419-044-5. Available from http://www.i6doc.com/en/livre/?GCOI=28001100817300.

to accomplish the goal; *num_crash* counts how many times the robot lies beyond a prefixed distance from an obstacle; *food_found* and *nest_reached* assume the integer values 0 or 1 and indicate whether the robot has reached respectively the food or the nest. M_1=0.3, M_2=0.3, M_3=0.2, M_4=0.1 and M_5=0.1 are constant weights and their sum must be equal to 1. These weights are chosen according to the relevance we want to assign to the parameters considered by the *fitness* function. Hence, fitness values will be in the range [0,1], where 1 is the worst result and 0 is the optimum. Our goal is to tune the *iCL* thresholds in order to balance the tradeoff among these performances measures.

First of all, we initialize the starting parameters with a plausible setting. Then, the DE algorithm starts to produce an initial generation G_0, of *NP* individuals, by randomly choosing the values in an unbounded space for the provided starting parameters. Since the interaction between the robot and the environment is not deterministic (due to the robot random movements), the same parameters combination can lead to different fitness value. Therefore, for each individual we calculate the average fitness on *m* runs. At the end of the *m*NP* simulations the algorithm selects the best fitness value for the global experimentation and a local best fitness value for the current generation. This process is repeated for *GEN* generations.

3.2 **Results evaluation**

In this section we report the results obtained by an experiment with *NP*=30 individuals, *GEN*=37 generations, *m*=10 repetitions, *F=0.85* and *CF=0.9*. In Fig. 5(a) the evolution of the global best fitness (solid line) and of the local best fitness (dashed line) obtained at each generation is displayed. Notice that, the best fitness starts decreasing after few generations. In Fig. 5(b) we also show the evolution of the AVOID AIRM-net *iCL* threshold values in the case of p_{bmax}=8.



(a) (b)

Fig. 5: (a) Global/local best fitness; (b) iCL thresholds of the Avoid-net

Let highlight that, following the DE algorithm, the parameters assume values in a wide range. In the description of the AIRM-net we claimed that the thresholds of neurons *iCL* must be in ascending order. While, at the beginning, the DE algorithm randomly selects the thresholds values, at last, we find that the best fitness value is generated by *iCL* threshold values, which are again in an ascending order, coherently

ESANN 2011 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 27-29 April 2011, i6doc.com publ., ISBN 978-2-87419-044-5. Available from http://www.i6doc.com/en/livre/?GCOI=28001100817300.

with the logic implied in our net. In the AVOID AIRM-net the role played by the *iCL* thresholds is to appropriately filter the $\Delta\sigma_t/p_\beta$ values provided by InI_t (see Fig. 2) in order to opportunely modify the clock period. Very small values of the *iCL* thresholds imply that for small variations of $\Delta\sigma_t/p_\beta$ all the *iCL* neurons fire and then the period p_β immediately turns to 1 (similarly to the classical architectures). Very high thresholds values imply a less sensitivity to minor changes (more similar to an architecture with fixed periodic activations). A high fitness value is observed in both cases: in the first case because of the increase of the *avoid_count* value (see Fig. 5(b) *GEN*=27) and in the second case due to an increase of the *num_crash* value (the sensors are checked only from time to time). In order to reduce both these performance measures, thus to minimize the fitness value, we have to balance the trade off between sensitivity and periodicity by choosing uniformly distributed thresholds. In our experiment we effectively observe that a best fitness is obtained when the *iCL* threshold values are distributed in the range of values assumed by $\Delta\sigma_t/p_\beta$, once fixed the environment.

4 Discussion

In this paper, we proposed a neural net implementing a mechanism of periodical and adaptive activation of a robot perceptual schema, able to deal with real time applications. Moreover, in order to make this net general purpose, we employ an evolutionary approach called Differential Evolution (DE). Among different evolutionary algorithms, such as Genetic Algorithms or Particle Swarm Optimization, we considered DE, since it allows, as we have noticed in the results, to explore a range of values that is not initially restricted. The results obtained by the automatic tuning of the neuron thresholds related to the AVOID behavior are very promising. Starting from these results we intend to test our model by extending the tuning also to the thresholds regulating all the behaviors of the architecture.

References

[1] E. Burattini, S. Rossi, *Periodic Adaptive Activation of Behaviors in Robotic System*, IJPRAI - Special Issue on Brain, Vision and Artificial Intelligence 22 (5) pp. 987-999, 2008.

[2] E. Burattini, A. Finzi, S. Rossi, and M. Staffa. *Monitoring strategies for adaptive periodic control in behavior-based robotic systems*. AT-EQUAL, pp. 130–135, 2009.

[3] E. Burattini, M. De Gregorio and S. Rossi *An Adaptive Oscillatory Neural Architecture for Controlling Behavior Based Robotic Systems*, Neurocomputing, 73 pp. 2829–2836, 2010.

[4] R. Storn and K. Price. *Differential evolution - a simple and efficient heuristic for global optimization over continuous spaces*. Journal of Global Optimization, 11 pp. 341–359, 1997.

[5] M. A. Arbib, *Schema Theory*, The handbook of brain theory and neural networks, MIT Press, Cambridge, MA, USA, 830-834, 1998.

[6] E. Burattini, A. De Francesco, M. De Gregorio, *NSL: A Neuro-Symbolic Language for a Neuro-Symbolic Processor (NSP).* IJNS. vol. 13, pp. 93-101, 2003.

[7] B. Gerkey, R. Vaughan, and A. Howard, *The player/stage project: Tools for multi-robot and distributed sensor systems*, in Proc. ICAR 2003, pp. 317–323, 2003.

[8] R. Brooks, *A robust layered control system for a mobile robot*. Robotics and Automation, IEEE Journal of Robotics and Automation, Vol. RA-2, No. 1, pp. 14-23, 1986.