2005 International Symposium on Nonlinear
Theory and its Applications (NOLTA2005)
Bruges, Belgium, October 18-21, 2005

Learning efference in CNNs for perception-based navigation control

P. Arena†, L. Fortuna†, M. Frasca†, D. Lombardo† and L. Patané†

†Dipartimento di Ingegneria Elettrica Elettronica e dei Sistemi, Università degli Studi di Catania
Viale Andrea Doria 6, 95125, Catania, Italy
Email: parena, lfortuna, mfrasca, dlombardo, lpatane@diees.unict.it

Abstract—Action-oriented perception involves complex tasks to be fulfilled in real time. In fact living beings, even the most simple, suitably integrate afferent stimuli, create an abstract, concise representation of environment stimuli and choose it for action-selection purposes. We propose a novel infrastructure, based on CNNs, where the spatial-temporal solutions are linked to the results of the perception stage. In this perspective, a primary role is devoted to the introduction of plasticity to enhance the association stimuli-CNN dynamics-action selection with application to the task of autonomous navigation control.

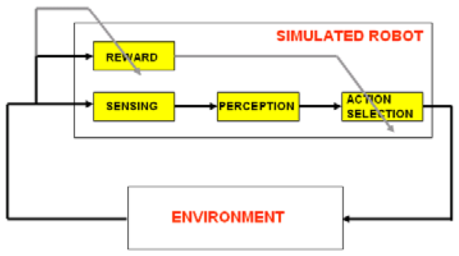

Figure 1: Functional block diagram of the implemented framework.

1. Introduction

It is known that an organism is able to perceive a set of simple sensory events (US, unconditioned stimuli), each of which automatically triggers a response (UR, unconditioned response) by the nervous system. According to *Classical conditioning*, [1], the repeated simultaneous presentation of an initially neuter stimulus (CS, conditioned stimulus) and an US is able to build an association between the two stimuli. This allows, after a number of trials, the CS to be able to command a response (CR, conditioned response), similar to the UR. *Operant conditioning* ([2], [3]), provides the animal with a further improvement in behavior by means of a task-dependent combination of rewarding successful actions and punishing unsuccessful ones.
Recently, on the basis of the new behavioral-based robotics paradigm [4] and of the neurobiological cues, machine perception research developed several bio-inspired frameworks of perception process, such as models of classical and operant conditioning [5] or a closed loop anticipatory network avoiding reflexes [6].
In this paper we present a new framework for the sensing-perception-action cycle emulating the low-level reflex reactions with the progressive structuring of a higher-level behavior. This strategy is applied to the simulation of a robot in a random foraging task.

The perceptive structure can be divided into functional blocks. Firstly we define a sensing block (afferent layer), which receives sensorial stimuli from the environment and sets the initial conditions for a two layer RD-CNN, which is the core of the perception process. The CNN parameters are chosen to generate Turing patterns, regarded as a kind of internal state for the whole system reflecting the state of the environment. Each pattern is associated with an action (efferent layer) by means of a traditional Motor Map (MM) [7]. The Reward Function (RF) plays a key role for the success of the whole strategy. Until now, it was selected *a priori*, based on design considerations. In this paper, the RF is not defined *a priori*, but is progressively learned by means of the association between simple and complex sensory events. Unlike classical conditioning, every US drives the learning of all the CSs. Once a basic reward function is formed, the reinforcement learning provided by the MM allows the robot to optimize the behavior in relation to the given task, according to the experiments in [2] and [3].
Moreover, a basic difference of our framework in comparison with [5] and [6] is the introduction of a dynamics in the system implementing the sensing-perception-action. Nonlinear dynamical systems are used in place of a static neural network, for reasons of biological plausibility and much improved plasticity. In this paper the sensing-perception-action loop is modelled by using nonlinear dynamical systems like CNNs, exploiting their real-time implementation [8].

2. The implemented framework

The implemented framework is made up of four main blocks (Fig.1): the sensing block, which receives environmental stimuli; the perception block, which forms an internal state from sensor input; the action selection block, which triggers an action to the effectors; the Reward Function (RF) block, which evaluates the effectiveness of the actions and contributes to the learning process. In the following we will refer to *iteration* to indicate the set of operations leading to a single robot action.

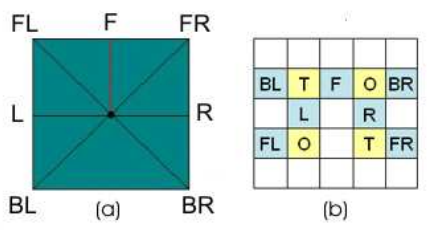

Figure 2: (a) Position of the obstacle distance and contact sensor within the robot (Front, Left, Right, Front-Left, Front-Right, Back-Left, Back-Right). (b) Initialization of the CNN first layer cells. In blue the cells set by obstacle distance stimuli (F, L, R, FL, FR, BL, BR) and in yellow those set by target stimuli (T,O represent respectively the target distance and orientation sensor).

2.1. The sensing block

The robot is assumed to have a squared shape. It is equipped with low level sensors (providing USs) and high level sensors (providing CSs). The low level sensors are: seven contact sensors (front, front-left, front-right, left, right, back-left and back-right as in Fig. 2.a); one proximity sensor for detecting the target.
The high level sensors are: seven distance sensors for the detection of obstacles, positioned on the robot as the contact sensors (Fig. 2.a); one distance sensor for detecting the target; one orientation sensor that determines the angle between the robot orientation and the direction robot-target.
All the sensors have a limited range and the obstacle distance and contact sensors have also a limited angular resolution. The low level sensors have a digital output: 1 if an obstacle (or a target) is present in the sensor range, 0 otherwise. The high level sensor output (*out*) is defined in pixels (distance sensors) or in degrees (orientation sensor). It is scaled in the range [0 1] by:

$$y(out) = e^{-\gamma \cdot out} \tag{1}$$

where γ defines the slope of the function. As an example, for a distance sensor i, the scaled output $y_{obs,i} = 0$ means infinite distance, whereas $y_{obs,i} = 1$ stance for null distance.

2.2. The perception block

The scaled output of each high level sensor sets the initial conditions of one or more cells in the first layer of the CNN (Fig. 2.b). This is the dynamical core of the perception block. We use a two-layer 5x5 RD-CNN ([9]) with zero-flux boundary conditions and appropriate parameters to generate Turing patterns [10], [11] and [12]. Each scaled distance sensor output is connected, in a topological way, to one of the cells of the first layer of the CNN setting its initial condition. The scaled output of both the target distance sensor and the orientation sensor fixes the initial condition of two cells of the first layer for sake of symmetry. The state of the cells of the first layer that are not connected to a sensor and the cells of the second layer were initialized to 0.
At each iteration we reset the CNN, initialize its cells again according to sensor outputs, and let the CNN evolve towards a stable Turing pattern.
Using the stable states of a multidimensional nonlinear dynamical system, like the CNN in the Turing patterns configuration, is an important mean to fuse and appropriately store a large amount of information, like that coming from a complex environment. Thanks to the parallel processing capabilities of CNNs [8], this kind of sensor fusion is not time-consuming.

2.3. The RF block and the action selection block

This block is responsible for the evaluation of the success or successless of an action, which drives the unsupervised learning algorithm implemented by a motor map. There is no *a priori* knowledge about the RF: it is only assumed to have a linear shape and depends on the output of the high level sensors (i.e. CSs):

$$RF = -\sum_i k_i \cdot y_{obs,i} - h \cdot y_{tar} - p \cdot y_{or} \tag{2}$$

where $y_{obs,i}$ are the scaled outputs of the obstacle distance sensor i, y_{tar} is the scaled output of the target distance sensor, y_{or} is the scaled output of the orientation sensor. The weights k_i, h ed p are randomly initialized in the range [-0.01 0.01].
First of all, it is necessary to learn the weights of the RF, i.e. the different meanings to be attributed to the different CSs. Such an attribution is performed by the association between the CSs and the USs. In original classical conditioning a specific CS is associated with a specific US. In our framework we regard the USs as anonymous sensors and each US can drive the learning of all the CSs. This assures much more plasticity and robustness: if a contact sensors would break down, the learning of the related obstacle sensor would not be compromised.
In a first phase the robot is provided only with a basic explorative behavior and reflex reactions (the URs). When there are no incoming USs and the action selection block is not yet active, an explorative behavior is triggered. It consists only in a forward movement of the robot.
When a collision occurs, i.e. an aversive US is sensed, a reflex reaction is automatically triggered for escaping from the obstacle. Such a reaction is different according to which contact sensor has revealed the collision. When the proximity sensor reveals a target, i.e. an appetitive US is sensed, a reaction toward the target is commanded. At each iteration, for the high-level sensors we evaluate a parameter S, called the distance-dependent sensor sensitivity, defined as:

$$S(t) = \frac{y(t) - y(t-\Delta t)}{\Delta t} \cdot \ max(y(t), y(t-\Delta t)) \tag{3}$$

When an US is sensed, for every sensors we calculate the average sensitivity of the last five iterations:

$$\widehat{S(t)} = \sum_j S(t - j\Delta t), \quad j = 0, .., 4 \tag{4}$$

Then, the RF weights $m = k_i, h, p$ are updated in the following way:

$$m^{new} = m^{old} + c \cdot \widehat{S(t)} \tag{5}$$

where the parameter c is defined as:

$$\begin{cases} c = 1, & \text{collision occurred;} \\ c = -10, & \text{target found.} \end{cases} \tag{6}$$

Such a different weight is necessary in consideration of the disproportion between the number of obstacle sensors and the number of target sensors. To avoid unbounded explosion of the weights value, after 10 iterations without any US occurrence, a decay rate is introduced: $m^{new} = 0.9999m^{old}$. Once learned, the RF should have positive k_i weights and negative h and p summarizing in a single value information about obstacle/target distance, and about the robot orientation towards the target. The higher the value for the RF, the better the situation for the robot, i.e. far from obstacles and near a target.
After a prefixed number of RF weight updates, the robot is allowed to learn the association between each emerged Turing pattern and a specific motor action by means of a MM supported by the RF described above. The output of the first layer cells of the CNN is connected, via synaptic weights w_r^{in}, to the neurons of a lattice A, each of which is associated with an output vector w_r^{out} which specifies an action. Nevertheless, the actual action performed by the robot, u is formed by adding a small random noise to the output weights. The weights w_r^{in} and w_r^{out} are randomly initialized, i.e. the association pattern-action is initially random. When an action is performed, the new incoming CSs (which represent the environmental feedback) set the current value of the RF. Such a value is compared with the previous one:

$$DRF(t) = RF(t) - RF(t - \Delta t) \tag{7}$$

A positive (negative) value for DRF indicates a successful (unsuccessful) action. Successful actions are followed by reinforcements like in the experiments described in [2] and [3] according to the following algorithm:
– let the CNN evolve and stabilize a Turing pattern;
– determine the winner neuron s, i.e. that one whose synaptic weight vector w_s^{in} best matches the input v;
– update the synaptic weights: $\Delta w_r^{in} = w_r^{in} + \varepsilon h_{rs}(v - w_r^{in})$
– update the output weights: $\Delta w_r^{out} = w_r^{out} + \varepsilon' h'_{rs}(u - w_r^{out})$.
The parameters ε and ε' are the learning rates, while h_{rs} and h'_{rs} are the interaction functions which allow to update also the weights of the nearest neighbors of the winner. Further details on the MM learning can be found in [12].

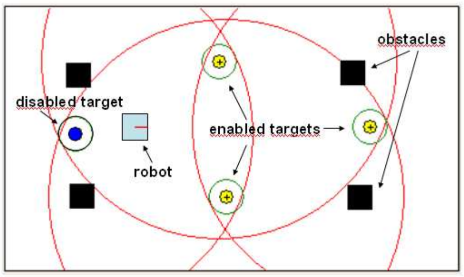

Figure 3: Simulation environment: for each real target both the region in which it is visible by the distance and by the proximity sensors is reported.

3. Results and discussion

The simulated environments, which the robot is placed in, are made up of obstacles, walls (that are considered as obstacles too) and targets. When the robot finds a target, this one is disabled and cannot be seen by the robot, even if the target is within the robot detection range. It is enabled again when the robot finds another target, that is in its turn disabled. This mechanism allows the robot to visit different targets. The training environment is shown in Fig. 3. The applied experimental protocol is made up of two phases:

1. Learning of only the RF weights: during this phase the robot, by the algorithm described above, the robot learns the rewarding mechanism for each high level sensors.

2. Refinement of RF weights and learning of the MM: the actions, initially randomly chosen, are then associated, via the MM unsupervised learning algorithm, with particular patterns so as to maximize the RF.

The robot start the experiment with only the reflex reactions behavior which allows the robot simply to rebound when a collision occurs and reach a target only within a small region around a target itself. When the robot suffers collisions or finds targets, the RF weights are updated. After a time of high oscillation in the weight values, such weights will differentiate, i.e. the robot is able to distinguish the meaning of the different CSs thanks to the association with the USs. At this point, second phase can start: while the RF weights continue to evolve and start stabilizing, the reflex reactions level is supported by the higher level behavior which is progressively learned by means of the unsupervised learning of the MM.
Fig. 4 shows the evolution of the RF weights during the overall simulation. It should be noticed that the RF weights related to the obstacle distance sensors converge to positive values, while the RF weights related to the target distance and orientation sensors stabilize on negative values.
This result outlines that the introduced algorithm for the right MM learning successfully distinguished the meaning of the two different kinds of sensors. Taking in account the

structure of the RF described above, the robot considers a high scaled output of an obstacle sensor as a negative situation, and a high value coming from a target sensor as positive meaning. Furthermore, the weight related to the obstacle distance sensor located on the frontal part of the robot (F) becomes far bigger than the other obstacle weights. Among the target sensors, the distance sensor reaches a value greater than the other, which could be pruned. These considerations can lead to consider this framework as a test useful for choosing the kinds of sensors to use on a robot and the positions where it is more convenient to locate them. The MM supported by the learned RF allows the robot to learn a higher-level behavior. The association between each emerged Turing pattern and the correspondent action guarantees a high level of plasticity in the choice of the action to perform due to the enormous number of Turing patterns that can emerge from a 5x5 RD-CNN. These patterns can represent many possible conditions of the environment and the unsupervised learning of the MM permits the robot to associate to each pattern the correct action as also demonstrated in [12].

4. Conclusions

A new framework has been presented for action-oriented perception in roving robots provided with different kinds of low level and high level sensors. The RF learning algorithm was proved to be very effective, allowing the robot to distinguish, without any previous knowledge, the different high level stimuli. Once learned the correct RF form, the MM, as already shown in [12], is able to construct a higher-level behavior. Future developments of this work include: a pruning of the low-weighted high level sensors, the definition of a more precise stop criterium for the RF learning algorithm. Furthermore, we will increase the number and the types of sensors for reasons of biological plausibility and in order to verify the sensor fusion allowed by the framework. All these developments are oriented to a future hybrid analog/digital hardware implementation.

Acknowledgments

The authors acknowledge the support of the European Commission under the project FP6-2003-IST2-004690 SPARK *Spatial-temporal patterns for action-oriented perception in roving robots*.

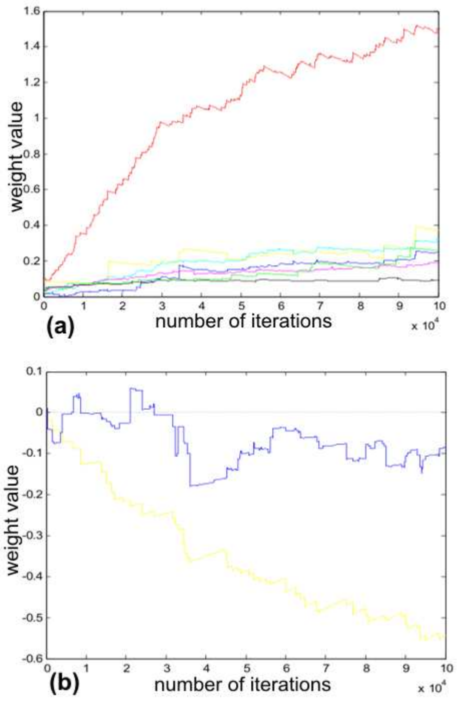

Figure 4: (a) Evolution of the RF weight related to the obstacle sensors: F(red), FR(violet), FL(cyan), R(black), L(blue), BR(green), BL(yellow). (b) Evolution of the RF weights related to the target sensors: target distance sensor (yellow), target orientation sensor (blue)

References

[1] I. I. Pavlov, *Conditioned Reflexes*, Translated by G.V. Anrep, London: Oxford University Press, 1927.

[2] E. L. Thorndike, “A constant error in psychological ratings”, *Journal of Applied Psychology* vol.4, pp.469–477, 1920.

[3] B. F. Skinner, *About behaviorism*, Alfred Knopf, 1974.

[4] R. C. Arkin, *Behaviour Based Robotics*, MIT Press, 1997.

[5] P.F.M.J. Verschure, T. Voegtlin, R.J. Douglas, “Environmentally mediated synergy between perception and behaviour in mobile robots”, *Nature* vol.425, pp.620–624, 2003.

[6] B. Porr, F. Wörgötter, “Isotropic Sequence Order Learning”, vol.15, pp.831–864, 2003.

[7] K. Schulten, *Theoretical Biophysics of Living Systems*, available in www.ks.uiuc.edu, 2002.

[8] R. Carmona, F. Jimnez-Garrido, R. Domnguez-Castro, S. Espejo, A. Rodrguez-Vzquez, “Bio-inspired analog VLSI design realizes programmable complex spatio-temporal dynamics on a single chip”, *Proc. 2002 Conf. Design Automation and Test in Europe*, Paris, France, pp.362-366, 2002.

[9] G. Manganaro, P. Arena, L. Fortuna, *Cellular Neural Networks: Chaos, Complexity and VLSI Processing*, Springer-Verlag, 1999.

[10] A.M. Turing, “The chemical basis of morphogenesis”, *Phil. Trans. Roy. Soc. Lond. B*, vol.237, pp.37–72, 1952.

[11] J.D. Murray, *Mathematical Biology I: An Introduction* (3rd Ed.) Springer-Verlag, New York, 2002.

[12] P. Arena, L. Fortuna, M. Frasca, R. Pasqualino and L. Patané “CNNs and Motor Maps for Bio-inspired Collision Avoidance in Roving Robots”, *CNNA2004*, 2004.