ELSEVIER

Robotics and Autonomous Systems 20 (1997) 225–250

Robotics and Autonomous Systems

Living in a partially structured environment: How to bypass the limitations of classical reinforcement techniques

P. Gaussier [a,*], A. Revel [a,1], C. Joulain [a], S. Zrehen [b,2]

[a] *ETIS–ENSEA/Cergy University, 6 Av. du Ponceau, 95014 Cergy Pontoise Cedex, France*

[b] *Center for Neural Engineering and Robotics Laboratory, University of Southern California, Los Angeles, CA 900089-2520, USA*

Abstract

In this paper, we propose an unsupervised neural network allowing a robot to learn sensory–motor associations with a delayed reward. The robot task is to learn the "meaning" of pictograms in order to "survive" in a maze. First, we introduce a new neural conditioning rule probabilistic conditioning rule (PCR) allowing us to test hypotheses (associations between visual categories and movements) during a given time span. Second, we describe a real maze experiment with our mobile robot. We propose a neural architecture overcoming the difficulty to build visual categories dynamically while associating them to movements. Third, we propose to use our algorithm on a simulation in order to test it exhaustively. We give the results for different kinds of mazes and we compare our system to an adapted version of the Q-learning algorithm. Finally, we conclude by showing the limitations of approaches that do not take into account the intrinsic complexity of a reasoning based on image recognition.

Keywords: Neural networks; Unsupervised learning; Topological maps; Reinforcement; Autonomous robot; All-or-none learning

1. Introduction

Nowadays, mobile robots are able to perform complex previously learned or predesigned tasks. For instance, they can follow a corridor until they find a specific identifiable scene which can be used either for recalibration of their internal map of the world or for deciding to perform a particular movement like entering a room or grabbing a given object. A strong effort has been put on the design of such control architectures allowing to react quickly by the use of some reflex mechanisms and at the same time by enabling them to plan longer term actions [5,8]. However, most of these systems do not have the ability to learn autonomously. Generally, robot learning needs two successive phases: off-line categorization of the possible situations and on-line learning of the associations between the categories and the actions. Categorization is often performed according to statistical methods (classification) or even designed by hand (a priori choice of thresholds on the sensor values).

* Corresponding author. E-mail: gaussier@ensea.fr.

[1] E-mail: revel@ensea.fr.

[2] E-mail: zrehen@rana.usc.edu. Supported by the Swiss National Fund for Scientific Research.

PII S0921-8890(96)0006-8

The first well investigated case of reinforcement learning (both by simulation and by real experiments) is the one in which the reward or the fitness value is available at each time step [3,4]. Interesting results were obtained either with pre-defined categories [25,28] or with input information simple enough to be directly used for a sensory–motor association. This works best when sensory categories are linearly separable [31]. When the categories are not linearly separable, a "hidden layer" of neurons must be added between the input and the output of the system. If unsupervised learning is to be used, that layer can be a winner take all (WTA) group [26], an adaptive resonance theory (ART) architecture ([6] for an application), or a quickly learnable version of the Kohonen map ([29,30] for applications) which in essence, is a statistical classifier. The more the examples of a class are presented, the more neurons are used to represent the class on the map. Therefore, if an insufficient number of instances of a given class is presented to the network, that class will be "forgotten". This can have undesirable consequences for the robot. Taking all these factors into account, we have previously developed a neural learning algorithm: the probabilistic topological map (PTM), which allows one-shot learning while preserving locally the topology of the input patterns, through the use of binary weights and stochastic weight modifications (see Box 2 [14]).

We have designed a regular building block called PerAc (perception–action architecture, see Box 1) to construct huge unsupervised neural networks devoted to the complete control of autonomous robots. We use a bottom-up approach that consists in making more and more complex systems relying on simpler ones. For instance, as we have shown in a previous paper [15], an attention orienting reflex (focus of attention) can be a good starting point to build a robot which is able to explore and to analyze a visual scene. The image analysis mechanism is performed by a single PerAc block. Furthermore, we have also shown that the output of that neural building block can be used

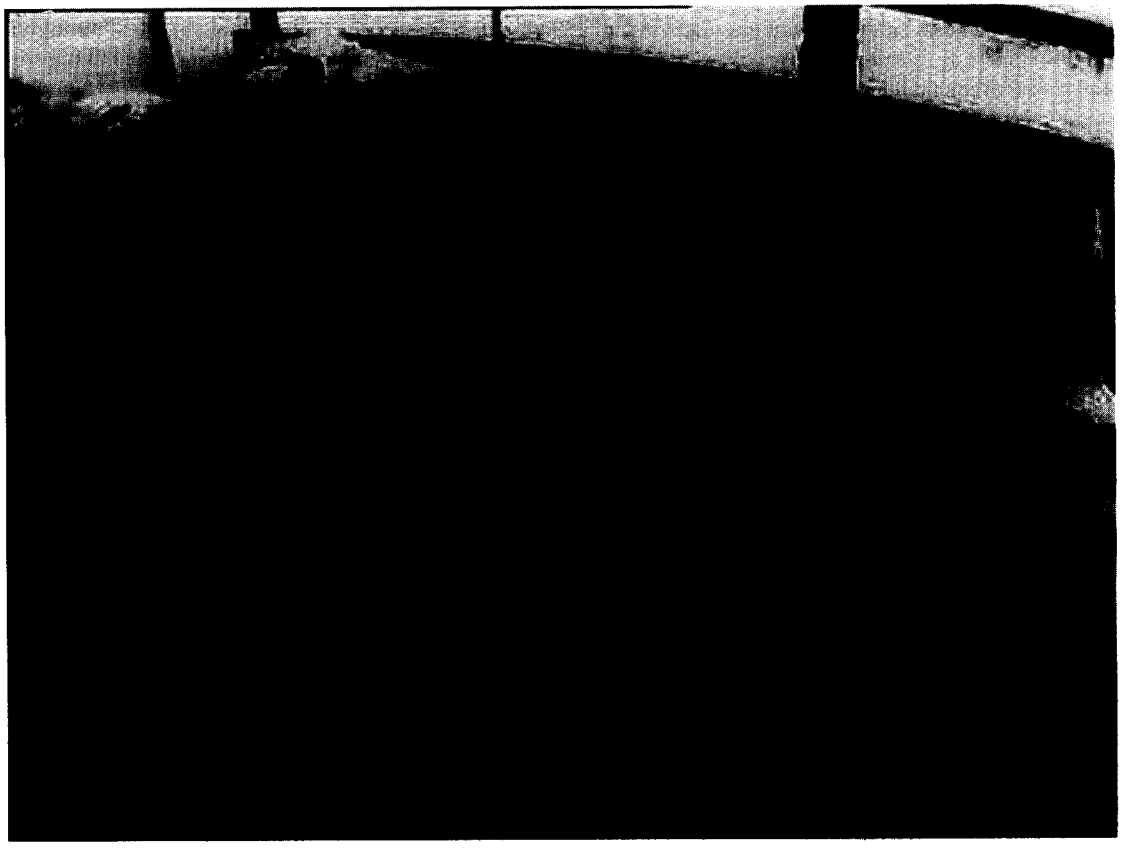


Fig. 1. Overview of a maze used in our experiments.

by a second PerAc block to learn to recognize a place in an indoor or outdoor environment and to learn how to reach it from any position in the surrounding environment [15]. However, in our previous systems, the robot was only able to perform instantaneous learning, such as associative learning or immediate reinforcement [11,14,15]. In this paper, we extend our previous on-line and immediate learning mechanisms to delayed reinforcement signals so that the robot will be able to learn a sequence of perception–action associations according to an internal or external reinforcement signal occurring later.

We chose to address the problem of "reaching a goal" in a partially structured indoor environment in order to avoid the multiple problems of identification present in a completely unstructured environment. In this paper, we present a study of how a robot can learn sensory–motor associations based on visual stimuli (obtained by a video CDD camera) in order to "survive" in its environment (Fig. 1). Practically, the robot's task is to learn the "meaning" of pictograms such as turning right when it sees a "right turn" arrow (see Fig. 7(b)). The task is defined with proscriptive constraints only [35] linked to internal motivations: we only define the viability domain of the system. More precisely, in our application, a pictogram can be learned because its association to a given movement will induce a delayed positive reinforcement signal (akin to the pleasure an animal feels when it eats something good) or an immediate negative reinforcement signal when the robot collides with a wall (equivalent to a pain signal in the animal context). The reinforcement signal can also be delayed if the robot does not succeed in solving a task after a given time. In the other cases, the robot will just learn repetitive or recurrent associations. It will only store really new patterns (different enough from previously learned patterns) and will have a low probability of changing its behavior.

In Section 2, we study classical methods for delayed conditioning and we introduce a new conditioning rule: probabilistic conditioning rule (PCR) allowing to test hypotheses (associations between visual categories and movements) during a given time span, afterwards the robot decides whether it will keep the same rule or not. Section 3 is dedicated to the description of a real maze experiment with a mobile robot. In Section 4, we propose to use our algorithm in a simulation in order to test it exhaustively.

2. Learning delayed sensory–motor associations

In this section, we present algorithms allowing a robot to make systematic sensory–motor (or stimulus–response) associations in order to find, in a maze, the shortest path from a starting point to a given goal location. The problem is to build a set of sensory–motor links which allow the use of immediate sensory information in order to perform actions leading to the goal. An important remark is that for a given set of sensory–motor associations, it is not known a priori if the set fits the problem well. The information is only available when the goal is reached or when a dead-end is encountered.

2.1. Neural network formalism of classical reinforcement techniques

The vocabulary of conditioning makes a distinction between a stimulus involving a reflex action of the robot (unconditioned stimulus – US) from any other kind of stimulus (conditioned stimulus – CS). Accordingly, it distinguishes a reflex response (unconditional response – UR) and a forced response (conditional response – CR). Conditioning can, therefore, be divided in two types, whether it is based on a UR or not (see [10,19,22,34] for a review).

In the first case, a link already exists between US and UR. The problem is then to reinforce the link between a CS and the UR. After learning, even if the CS is presented alone, the robot should produce the CR. Such simple conditioning mechanisms are based on the Hebb rule (see Eq. (1)) and on classical models of formal neurons. The weight of each link between neurons i and j is given by W_{ij}. The neuron activity O_j is defined by

$$O_j(t) = f\left(\sum_{i=1}^{N} W_{ij}(t) \cdot I_i(t)\right), \qquad W_{ij}(t+1) = W_{ij}(t) + \epsilon \cdot I_i(t) \cdot O_j(t), \tag{1}$$

where I_i terms represent the inputs, ϵ is the learning rate, and f a non-linear function. In the second case, no US is available. Therefore a global reward signal must be introduced to tell whether the association between the CS and the CR is correct or not [2,4]. Those models work quite well if the reward is given immediately after a correct or an incorrect action was performed. But, in most cases, and particularly for our task, the reward can only occur at the very end. So a response must be proposed without any US and the robot must reinforce the link between the CS and the CR only according to the delayed reward. This reward signal takes at least two opposite values in order to tell the robot if the association is "good" or not. To be able to find by itself which movement to perform, the robot has to be able to test different responses in order to choose which one should be associated to the CS. An intuitive solution consists in selecting a random response (by adding noise to the neuron output – see Fig. 4(a)) and testing the consequences. The number of stimuli and responses must not be too large, otherwise the probability of finding the right association will be very low (NP-complete problem [24]).

In that context, Barto and Sutton [2,4] propose an efficient solution using both a diversity generator (selection at random) added to the neuron activity (see Eq. (2)), and an equation adapted from the Hebb rule (see Eq. (3)):

$$Act_j = W_{0j} + \sum_{i=1}^{N} W_{ij} \cdot I_i + \text{noise}, \qquad O_j = \begin{cases} 1 & \text{if } Act_j > 0, \\ 0 & \text{otherwise}, \end{cases} \tag{2}$$

where Act_j is the neuron activity before threshold,

$$W_{ij}(t+1) = W_{ij}(t) + \epsilon \cdot [P(t) - P(t-1)] \cdot [O_j(t-1) - O_j(t-2)] \cdot O_i(t-1),$$

which is equivalent to

$$\frac{\partial W_{ij}}{\partial t} = \epsilon \cdot \frac{\partial P}{\partial t} \cdot \frac{\partial O_j}{\partial t} \cdot O_i, \tag{3}$$

where $P(t)$ is the reward signal at time t. $P(t)$ has a low value when the system does not solve the problem and a higher value when its behavior is better.

This last rule will be the starting point of our discussion even if it has been improved for years by trying to predict the reinforcement signal through time (for instance $TD(\lambda)$ and Q-learning).

2.2. *Presentation of problems linked to maze experiments*

In an ideal case, we can consider a maze as an environment divided into simple situations: corridors, T-junctions with a "turn left" arrow on the wall (called "pictogram"), T-junctions with a "turn right" arrow and dead-ends (see Fig. 2). At this point, we assume that wherever the robot might be in a corridor, it recognizes it as such (we will show in Section 3 that categorization is a complex problem and we will propose a neural network (NN) that dynamically allows this kind of categorization). At a T-junction, the robot learns to recognize the pictogram shown indicating in which direction it must turn. The robot movements are restricted to: "go straight", "turn left 90°" or "turn right 90°". In Fig. 2 for instance, the robot must reach the hammer located in the left branch of the maze. Then, it must learn to associate the recognition of a corridor to a "go straight" movement. When it reaches the T-junction, it is shown a "turn left" arrow. The robot may "turn right", then subsequently reach a dead-end, and receive a negative reinforcement signal. So, in the rest of the experiment, the robot should learn to turn left when it sees the "left arrow" instead of "turning right". Thus, it will obtain a positive reward which allows the learning of the correct set of perception–action pairs (see Fig. 5).

Most methods [23,38] do not take into account the fact that situations are not necessarily equiprobable. If we use a model in which noise is added to the output of the neuron, learning can be unstable. When the robot tests a set of sensory–motor associations and receives a negative reward, the global noise level grows. So the robot will move randomly with a high probability of performing a "stupid" movement in frequent situations without a high probability of choosing the correct action in rare situations. Moreover, these methods suppose that a place can be

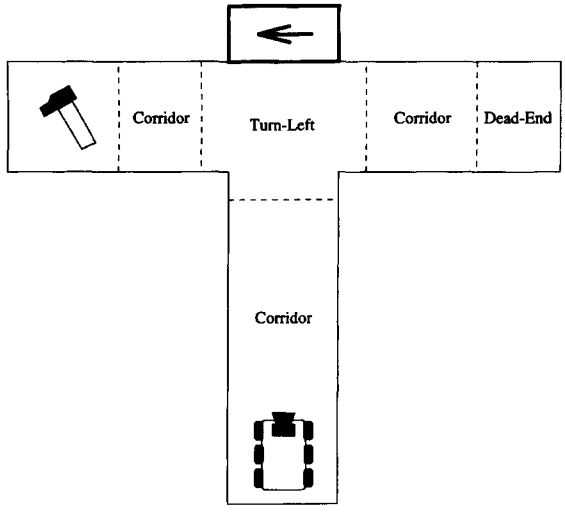


Fig. 2. The robot must reach the hammer dripped on the left-hand side of the maze. Dotted lines represent the boundaries of the perpetual categories.

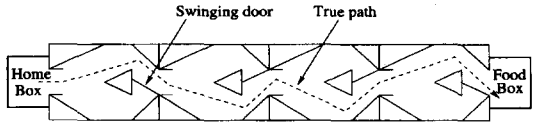


Fig. 3. Krechevsky's experiment: a rat is on the home box and must reach the food box. In each box the correct door may be determined by the experimenter in terms of its being light or dark, left or right.

easily recognized as such. For instance, when the robot moves in a corridor the visual information can change in such a perceptible way (see Fig. 11) that it cannot be categorized as a single state by an unsupervised recognition system. As a result, pictures must be taken at a rate high enough in order to avoid missing potentially important events. Therefore, we decided to build a conditioning model that allows to bring situations with different occurrence probability on the same level (PCR).

2.3. *Making and testing hypotheses: the PCR algorithm*

Studies of animal strategies in maze experiments provide good hints to design robot control architectures that solve problems noticed in Section 2.2. For instance, Krechevsky [20] has proposed an interesting experiment in which a random search is unable to explain the rat's capacity to solve a high dimension association problem. In his experiment, the rat must pass 10 times a day through four identical discrimination boxes (see Fig. 3). In each box there are two doors and the correct door may be determined by the experimenter in terms of its being light or dark, left or right. So, there are 40 choices to be tested. Krechevsky found that the rats mainly performed a systematic well-above-chance choice called "hypothesis" (see also [21,37] for all-or-none learning).

In this section, we propose a neural learning rule that tries to model such a behavior. The robot uses a hypothesis long enough to test its consequences and to decide if it needs changing. As global noise added to the output of a neuron leads to unsteady states, we choose to introduce diversity generators (noise) on each synaptic weight. Moreover, in the case of the simple sensory–motor associations encountered in maze experiments, weights do not

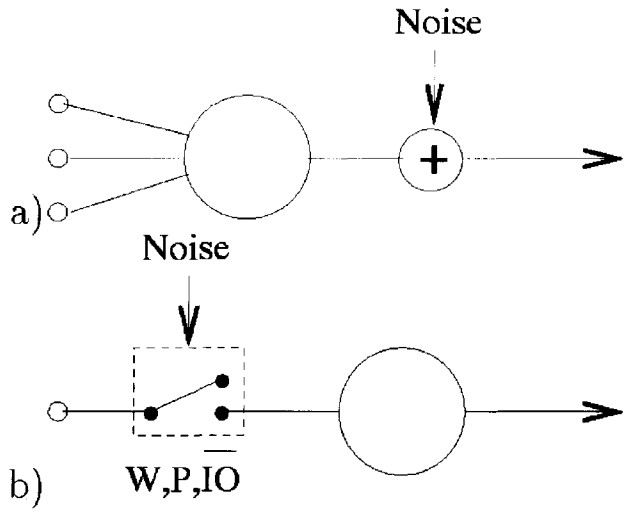


Fig. 4. (a) Classical neuron model with noise added at the neuron output. (b) Schema of the PCR model.

have to be analog: a binary value is enough to indicate the existence of an association between a particular recognition and an action. But we now come to contradictory requirements: how to modify binary weights while keeping the same hypothesis for long enough? Our solution is to introduce a probability associated to the binary weight which gives confidence to the input/output association it represents (see Fig. 4(b)). When a reinforcement signal occurs, only the probability term is changed. Yet, a random draw is done in order to change weights whose confidence term is low. Such a mechanism gets the robot to behave as if it was testing hypotheses. It is interesting to draw a parallel between the PCR algorithm and genetic algorithms (GA). The way binary weights are switched can be compared to the GA mutation process which allows bits change in the DNA sequence of an individual. Yet, while GA test the behavior of populations whose DNA is fixed, the PCR allows a similar process during the "life" time of a single robot (kind of evolution: the robot tests different configurations of its weights during the same "life"). In fact, our mechanism is much more similar to the neural-Darwinism proposed by Edelman [7,9] and the notion of re-entrant maps [32] (but here the link suppression can be performed according to a delayed fitness value).

In order to know how to modify the probability p_{ij} associated to a synaptic weight, an input–output correlation measure must be stored. At each time step, three (time-integrated) parameters, associated to the input I_i, the output O_j and the *input · output* product are computed. They are designated by: $\overline{I}$, $\overline{O}$ and $\overline{IO}$. These values can be computed in a neural fashion by averaging on a time window of a given width. Formula (6) is a way to compute a first-order integration in a time window of width $\tau + 1$.

Simplified PCR Algorithm

Activation rule:

$$Act_j = Max_i(W_{ij} \cdot I_i) + \text{noise}, \tag{4}$$

$$O_j = \begin{cases} 1 & \text{if } Act_j = \text{Max}_k(Act_k), \\ 0 & \text{otherwise.} \end{cases} \tag{5}$$

Updating at each time step: $\overline{I_{ij}}$, $\overline{O_{ij}}$ and $\overline{IO_{ij}}$ updated according to the following equation:

$$\overline{X_j}(t+1) = \frac{\tau \overline{X_j}(t) + X_j(t)}{\tau + 1}. \tag{6}$$

If $|\partial P(t)/\partial t| > \xi$: *Probability updating*

$$\Delta p_{ij}(t) = \alpha \cdot \frac{\partial P}{\partial t} \cdot C_{ij} \cdot f_B\left(W_{ij}\right), \qquad p_{ij}(t+1) = p_{ij}(t) + \Delta p_{ij}(t) \tag{7}$$

with

$$C_{ij} = \overline{IO}_{ij} \Big/ \sqrt{\overline{I}_i \overline{O}_j}. \tag{8}$$

and

$$f_B(W_{ij}) = \begin{cases} 1 & \text{if } W_{ij} = 1, \\ -1 & \text{if } W_{ij} = 0. \end{cases} \tag{9}$$

If $Rnd > p_{ij}$ and $\overline{I} \cdot \overline{O} \neq 0$ then

$$W_{ij} = 1 - W_{ij}, \qquad p_{ij} = 1 - p_{ij}. \tag{10}$$

noise is a random value as little as wanted.
$P(t)$ is the global reinforcement signal. It represents a way to measure robot satisfaction over time.
α is the delayed conditioning learning rate.
ξ is a constant fixed by the experimenter.
Rnd is a random value in [0, 1].
$p_{ij} \in [0, 1]$, $W_{ij} \in \{0, 1\}$.

The input/output normalized correlation is given by Eq. (8). This term does not depend on "appearance" probabilities of input–output pairs. For instance, if a situation is encountered N times while another is only encountered once (both involving a given output each time they are met), in both cases the correlation C_{ij} is equal to 1. This normalized Hebbian term is very similar to the eligibility term used in Barto and Sutton [2,4].

Each time the reinforcement signal $P(t)$ varies enough ($|\partial P(t)/\partial t| > \xi$), confidence terms are updated according to formula (7). After updating the probabilities a random draw is done so as to determine whether a weight will be changed or not (see Eq. (10)).

A typical learning situation is proposed in Fig. 5. At the beginning ($t = 1$), the robot tries to "turn left" in the corridor and it collides with the wall. A negative reinforcement signal is emitted and the synaptic link between the "corridor" situation and the "turn left" movement is inhibited. Next ($t = 2$), the robot tries to turn right and the same thing happens. Finally ($t = 3$), it tries to go "straight ahead" and succeeds because it avoided a negative reinforcement. A positive reinforcement signal is emitted and the association between "corridor" and "go straight ahead" is learned. The robot will have no more problems with this situation and the problem complexity is reduced. When the robot meets the "left arrow" shape ($t = 5$ in Fig. 5), it will try to go straight ahead and will end up turning right by chance. The association is not painful and can be at that moment considered as correct. This is where the problem of delayed conditioning begins. The weight is not directly strengthened but the C_{ij} variable (correlation between input and output in Eq. (8)) is updated. When the robot reaches the dead-end, it receives a negative reinforcement. It modifies the probabilities associated to each weight according to the reinforcement signal and thus decreases the probability associated to the link W_{lr} between the "left arrow" recognition and the "turn right" movement. Finally, the probabilities are used to decide whether the weight must be switched to its complementary value and obviously the weight W_{lr} has a high probability to switch from 1 to 0.

In the worst case, if the robot receives increasingly negative reinforcement, the confidence associated with the activated links decreases and all the weights will end up switching to a null value with a confidence of 1. At that point, the PCR algorithm becomes equivalent to a random search (because of the little noise added to the neuron outputs). Thus we have set a lower bound to the algorithm convergence. By chance, in all our tests, the rough information provided by the reinforcement signal was large enough to ensure results better than random choices.

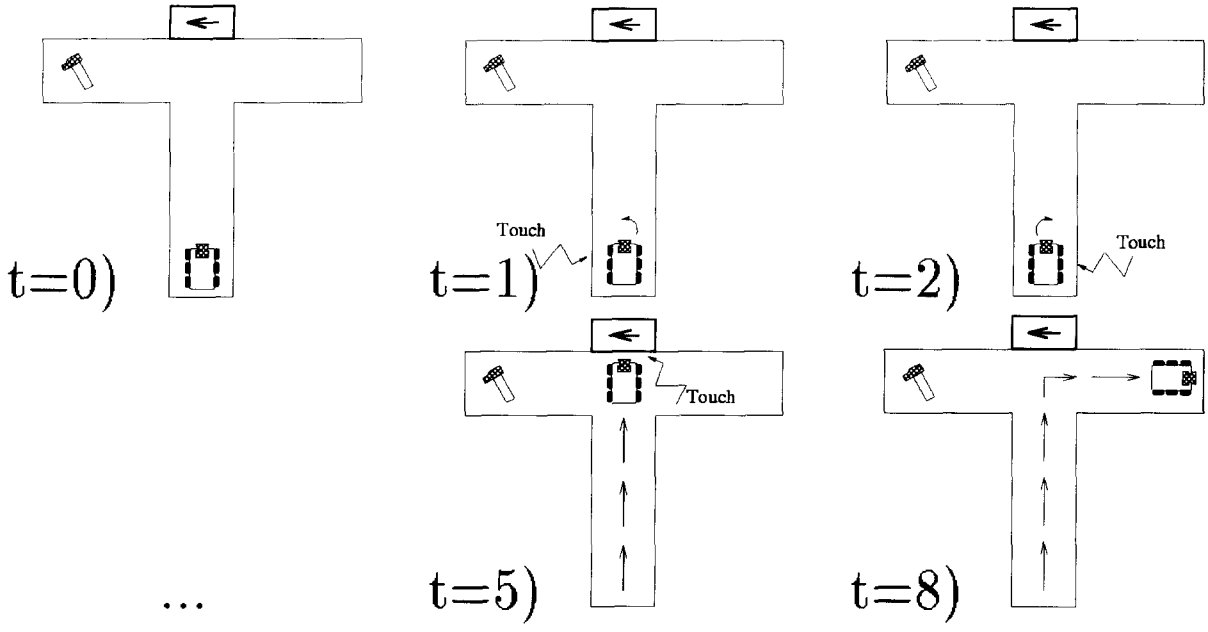


Fig. 5. Example of a course in the maze. At time $t = 0$ the robot is introduced in the maze and must reach the hammer dropped on the left-hand side of the maze. It tries several movements in the corridor until it learns to go straight ahead. When it arrives at the T-junction it turns right while it should have turned left. As it arrives in a dead-end, it will reconsider the movements it has performed.

2.4. *The reinforcement control system*

In this kind of experiments, the reinforcement signal can be either binary (the good set of sensory–motor associations has been found or not) or analog (the solution is more or less correct). In the first case, the robot must perform a random search because no gradient in the reinforcement signal is available. In the second case, PCR uses the reinforcement variation to reach the good solution quicker. On the other hand, we cannot accept that the robot gets to the end of the maze experiment before it stops scraping walls. If an immediate negative reinforcement signal is encountered, the robot must learn to avoid the wrong movement immediately.

Another problem can occur when the robot spends too much time without any negative or positive variation in the reinforcement signal ($\partial P(t)/\partial t < \xi$). Imagine a loop inside a maze. If the robot was mistaken on the interpretation of a pattern at a T-junction of the loop, it can turn round and round in this loop without any reinforcement (because the robot neither collides nor reaches a dead-end). To stop this dead-lock situation, the robot needs an internal signal which allows to change the associations it has made (such as a "hunger" or "I am fed up" signal). This mechanism is equivalent to testing several hypotheses during a single exploration. For the sake of space and clarity, the internal reward system (limbic system) will not be described in this paper (see [18] for more detail).

2.5. *Discussion on the PCR algorithm*

First of all, we chose binary weights in order to be compatible with the probabilistic topological map (PTM – see Box 2) which is used for on-line learning of the perceived situations. Moreover our interest is more in finding a minimalist delayed reinforcement conditioning rule than in optimizing a state space exploration mechanism. Accordingly, we refuse all the algorithm optimizations that contradict the local computation aspect of the neurons.

Second, in the description of the PCR algorithm, we have supposed that categories had already been built and that the PCR algorithm was used to connect these categories to the "good" outputs only. In a real neural architecture devoted to the control of an artificial animal, things are not so easy because in the beginning, no category exists, and the neurons to associate must be chosen out of thousands of other neurons. For instance, all the neurons of the $N \times M$ recognition map (typically more than 10×10 neurons on our PTM map) can be linked to any output movement. Theoretically, there are $3^{N \times M}$ ($3^{10 \times 10} = 5.15 \times 10^{47}$) possible associations. Moreover, we do not want the algorithm to destroy reliable associations learned in previous experiments. To fulfill that goal, only links between the winners of the map and the WTA are learned. This choice allows to limit the number of links that are actually learned. It also avoids confusion with the neuron being activated because it is closer or further from the winner input neuron (in the computation of the output neuron activity).

If we consider an exhaustive or random search algorithm, there is a little chance that it can find all the sensory–motor associations in a reasonable time because of the combinatorial explosion. Obviously, the search process could only be performed between situations that have been encountered during a set of random trial for instance. This supposes that these situations are kept in memory by a mechanism similar to our time correlation procedure (C_{ij} parameters). Also this brings a problem for the viability constraints of the robot. PCR is mainly designed to keep the robot from performing the same big mistakes again and again. Moreover, for an exhaustive search (enumeration) there is always a risk from one trial to another that the same situations will not be encountered. Then, the complete enumeration would have to start over again. Another solution might be to use a linked list to store all the new associations' possibilities that appear during the maze exploration and that have not already been tested. But in that case, the computational cost of the algorithm increases drastically. However, enumeration must be rejected because it does not rely on local computation. Accordingly, for the PCR rule, we could only modify one weight in each trial avoiding difficulties linked to the assignment of the reinforcement signal. But, this modification supposes the possibility of modifying the weight of the first neuron before modifying the weight of the next one. Neurons would need information coming from a supervisor to decide which neuron might try to modify its weights and that denies our local computation a priori. We must thus reject this optimization of the PCR rule that would also renounce the local computation of weight modification.

PCR is an algorithm which can be used in many applications depending on the kind of reinforcement signal which is provided. Moreover, our algorithm can be improved in order to use analog weights. We just need to take as an analog value the product of the probability p_{ij} and the binary value W_{ij}: $w_{ij} = p_{ij} \cdot f_B(W_{ij}) + 1$. The interest is then that this analog value cannot vary continuously, thereby avoiding the precision problems found in slow learning algorithms. Furthermore, it allows to maintain the same properties as the binary PCR algorithm and use the analog values (modulation of the activity). PCR can also be improved as follows to allow simple associative or Hebbian learning and to speed up its convergence time:

$$\Delta p_{ij}(t) = \left(\alpha \cdot \frac{\partial P}{\partial t} + \epsilon\right) \cdot C_{ij} \cdot f_B(W_{ij}) - \lambda \cdot p_{ij}(t) \cdot I_i(t). \tag{11}$$

$\epsilon \cdot C_{ij} \cdot f_B(W_{ij})$ allows us to keep a "Hebb rule" in order to make classical conditioning possible and $\lambda \cdot p_{ij}(t) \cdot I_i(t)$ allows us to avoid unstable behavior of the $p_{ij}(t)$ terms (pseudo-normalization of the p_{ij}).

Box 1: The PerAc architecture. The PerAc (Perception–Action) block has been proposed as an elementary generic brick of neuronal computation [1,5,17]. It allows on-line learning of sensory–motor associations. A PerAc block is divided into two levels corresponding to the action and the perception data flows. The first level is a reflex mechanism which extracts basic information from the perceived input so as to directly and roughly control the actions. The second level performs situation recognition and allows learning of the associations between what is recognized in the perceptive flow and the chosen movement. This level permits to maintain the behavior provided by the reflex system or to avoid it when there are contradictions with the robot's viability constraints. All perceived information as well as the motor output are expressed in egocentric polar coordinates. Thus direct "one-to-one" links between

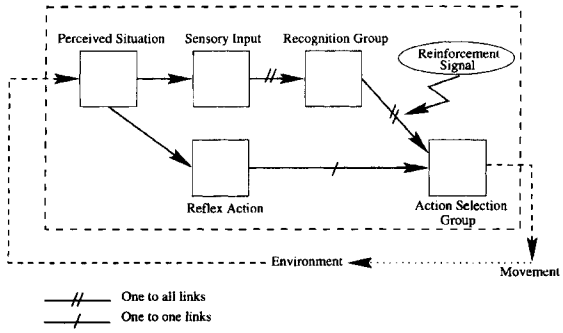


Fig. 6. Schematic representation of the PerAc block. From the perceived situation, the reflex system extracts information to control directly the actions. Concurrently, the recognition system learns sensory input patterns and how to link them to the action by associative or reinforcement learning. The system adapts itself dynamically to the environment.

the reflex action proposal group and the action selection group allow making movements in the direction of features already known as "relevant". A PerAc block is a competitive network composed of four neural boxes as seen in Fig. 6.

At the beginning, the robot has not yet learned anything. All the movements it performs are provided by its reflexes. Each time it perceives a "new" situation, it categorizes it. Depending on the learning procedure used to associate the recognition group with the action selection group, several different behaviors can be obtained. If the learning process is an associative rule (Hebb rule), the robot learns to associate the recognition of a situation with the action which is proposed by the reflex system. It thus allows to obviate the reflex system's deficiencies, and to overcome it if the US is absent. For instance, if in front of a given situation the reflex is unable to choose a response, the recognition system can generalize from learned perception–action schemes and trigger the most appropriate action. The use of reinforcement learning as an adaptive rule allows to make the system even more flexible. Indeed, if the reflex system provides a given action which is not adapted to the environment, a negative reinforcement signal can make the system learn to avoid this movement. The reflex action is then inhibited and replaced by another action which better fits the situation. Each action of the robot modifies its perception in a particular way. We try to have an appropriateness of the robot with its environment and we refuse to choose a priori the robot's categories (visual, motor, etc.). The interesting point is that all the robot's behavior is not completely coded in the network. In fact the system evolves because of the dynamical interaction between the robot and its environment.

3. Situation categorization and association learning in a real maze

3.1. Introduction

To test our global architecture applicability, we have developed a simple maze experiment with a prototype of the Koala robot (25 cm × 25 cm – see Fig. 7(a)) (Kteam – LAMI/EPFL). NN is computed on a SUN Sparc 5 workstation. Information is transmitted by a bidirectional serial link at 19 200 bauds. The global size of the maze is 3 m × 3 m and the corridor width is about 60 cm. Fig. 7(b) presents an example of an image taken inside the maze with the on-board standard CCD camera (the image is subsampled to 128 × 100 pixels).

Whatever the merits of some conditioning rules, if inputs are not linearly separable, they cannot be used directly to compute an action. The image complexity must be reduced by finding relevant information in the image. For this purpose, we have chosen to represent a given pictogram by a specific oriented texture. A "right turn" arrow has been chosen to correspond to a rectangle made of vertical stripes while a "left turn" is a rectangle with horizontal stripes. Shapes are detected by a series of Gabor filters. For legibility reasons we did not give details of the complete visual

Fig. 7. (a) The Koala robot. (b) Example of an image taken by the CCD camera.

system (for further explanation, see [11]). In fact, it is only important to understand that the visual system enables the robot to perceive the pictograms as if they were in the center of the image, even if in reality the images can be slightly translated or rotated. Practically, our robot's visual system includes a mechanism of attention focus which provides translation invariance.

An experiment consists of a series of trials which make the robot learn a given number of sensory–motor associations so as to obtain or to maximize a final reinforcement signal (robot goal). Each trial is a sequence of perceptions and actions that end by a reward or a punishment. A path is considered as a trial if the robot is at the starting point and leaves the maze at a dead-end or arrives at the goal. The robot's progress are computed in "time steps". A time step is a sequence of perception/computation/action. The reinforcement signals are provided by two electrical contacts allowing to measure the conductivity of the object the robot collides with. If the object is not a conductor, it is considered that the robot has collided with this object. It corresponds to an immediate negative reinforcement signal (like a shock). Conversely, if the object is a conductor it is assumed to be the goal. It corresponds to the positive reinforcement signal (reward).

3.2. Robot "brain" architecture

A simple PerAc block is used to control the robot (Fig. 8; see also Box 1 about the PerAc architecture). In our experiment, the reflex consists in following the corridor. The motor input (MI) box is a group of neurons that simply filters the input image so as to detect known relevant features in the image for corridor following (here the vanishing point of the corridors -- see Section 3.2.3). In a simplified case, if we do not take into account the corridor following problems, there are only three movements: turning left 90°, turning right 90° and going straight ahead. The motor output (MO) is a WTA group of three neurons intended to provide only one action at a time. A small noise is added to each of these neurons in order to avoid indecision when different neurons have the same activity.

The higher level is composed of two groups of neurons: the visual input (VI), the visual output (VO) interconnected by the mean of plastic links. VI consists of a series of Gabor filters which are able to detect four orientations of

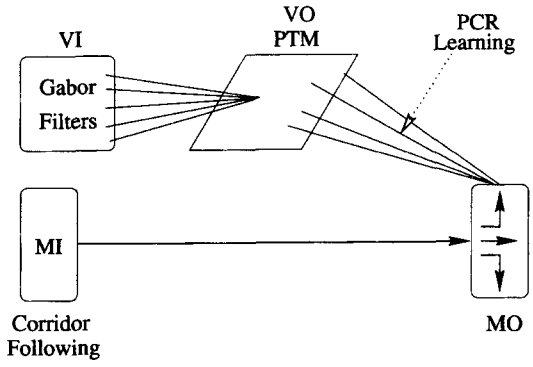


Fig. 8. The neural architecture used to realize the experiment of sensory–motor associations. VI = visual input (CCD camera), VO = visual output (recognition map), MI = motor input (reflex system) and MO = motor output (effectors).

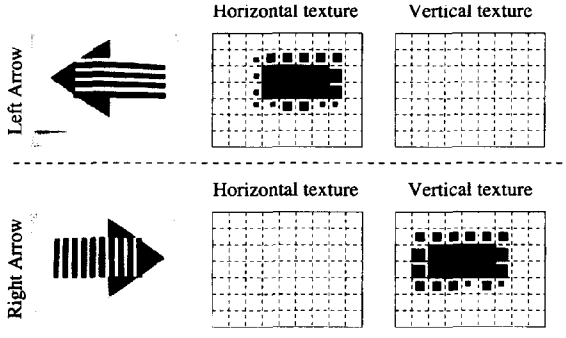


Fig. 9. Example of the detection of spatial frequencies on two maps with different orientations. The "turn left arrow" is represented by horizontal stripes, so, only the map sensible to horizontal orientations of spatial frequencies is activated. In the other case ("turn right arrow") it is the contrary.

three spatial frequencies in all the positions in the image. Each frequency/orientation pair corresponds to a neuronal map. There are thus 12 maps. If a neuron on these maps is activated, it means that a spatial frequency of a given orientation has been detected at the corresponding position in the image. A competition between the different maps for all the positions in the image guarantees there is only a winner frequency for each position.

VO is a probabilistic topological map (PTM) which codes patterns obtained in VI (see Fig. 9 for an example). This map provides the different categories that will be associated with a movement. Indeed, the weights between VO and MO are learned according to the PCR algorithm.

3.2.1. Topological map utilization for on-line categorization and association

Let us come back to our example (Fig. 5). We have presented in Fig. 10, the different images the robot sees when it moves along the corridor. The patterns formed by the horizontal stripes on the wall correspond to the "turn left" arrow. [3] It can be correctly recognized in situations c, d and e (Fig. 10) but not a and b.

At the beginning, the robot has no knowledge about the visual shapes it can encounter (except that they are textured). The vigilance term – a parameter of the PTM that determines the level of similarity to previously learned

[3] In fact it can be seen in the reflection on the ground that a real arrow has been drawn but as the image is overlighted, it cannot be seen. Besides, it does not matter because, due to Gabor filtering, the robot can only recognize spatial frequencies.

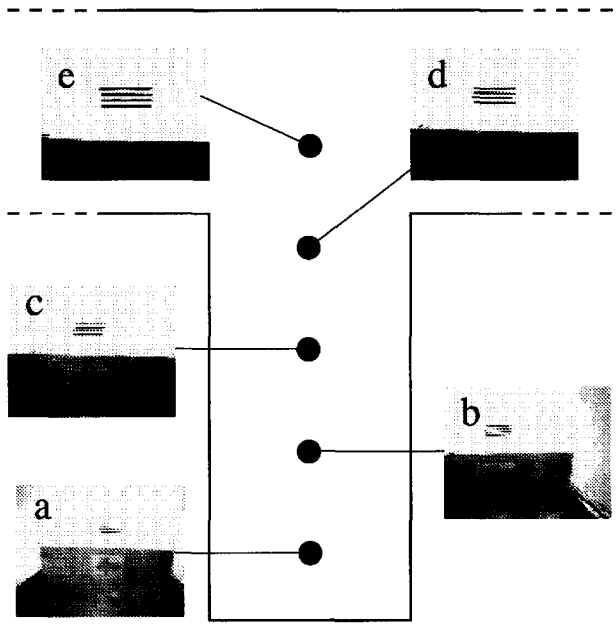


Fig. 10. A sequence of images taken during the movements of the real robot into the maze. The position of the snapshots into the pictures corresponds to the location where the robot was when it took the image. The starting position corresponds to situation a. Situations b, c and d correspond to "straight ahead" movements. When the robot arrives in situation e, it is just in front of the perpendicular corridor.

patterns required to learn a new pattern (see Box 2 and Appendix A) – has a medium value (practically $\rho = 0.7$). When the robot sees the situation a, it codes it on the neuron C_1 of the PTM map (VO group) which has won (see Fig. 11). As weights have an initial random low value, any neuron can be the winner. In the VO group, weights of the neurons surrounding the winner neuron are modified with a probability which depends on the Euclidean distance of these neurons to the winner. This mechanism allows us to make neurons in the winner's neighborhood react to a pattern similar to the one it has coded. Furthermore, an activity diffusion bubble is also created around the winner. As no link between the VO and MO exists yet, a movement is chosen randomly, according to the noise. If it is "turn 90° left or right", the robot collides with the wall. This collision activates a negative reinforcement signal which is used in the PCR algorithm to decrease the probability of association between the situation a and one of the turning movements. Yet, there is still no link between neuron or category C_1 and any movement. The subsequent movements are thus still chosen randomly until the robot chooses the "straight ahead" movement. At this moment, the movement being successful, the negative reinforcement stops and a positive reinforcement signal is emitted ($\partial P(t)/\partial t > 0$). A link between category C_1 and the movement "straight ahead" is learned and its associated probability is increased.

After moving ahead one step, the robot sees the image b. As it is similar to the previous one,[4] it activates a neuron close to C_1. Due to the diffusion bubble, C_1 is also activated but with a lower value. There is a link between C_1 and the "straight ahead" action, so this movement is performed (see Fig. 11(c)).

From situation c to e (Fig. 10), as horizontal stripes begin to appear increasingly clearly, they are detected by Gabor filters. This activates a bubble which is further away from C_1. As the vigilance term is rather low (see Box 2 and Appendix A), the system tends to overgeneralize and it continues performing the same movement. As the robot arrives at situation e, it tries to move ahead but collides with the wall. Then, a reflex system makes it move

[4] As Gabor filtering is performed, almost no spatial frequencies are detected, neither in image a nor in image b. So the pattern of the filtered image corresponds to a vector with null components.

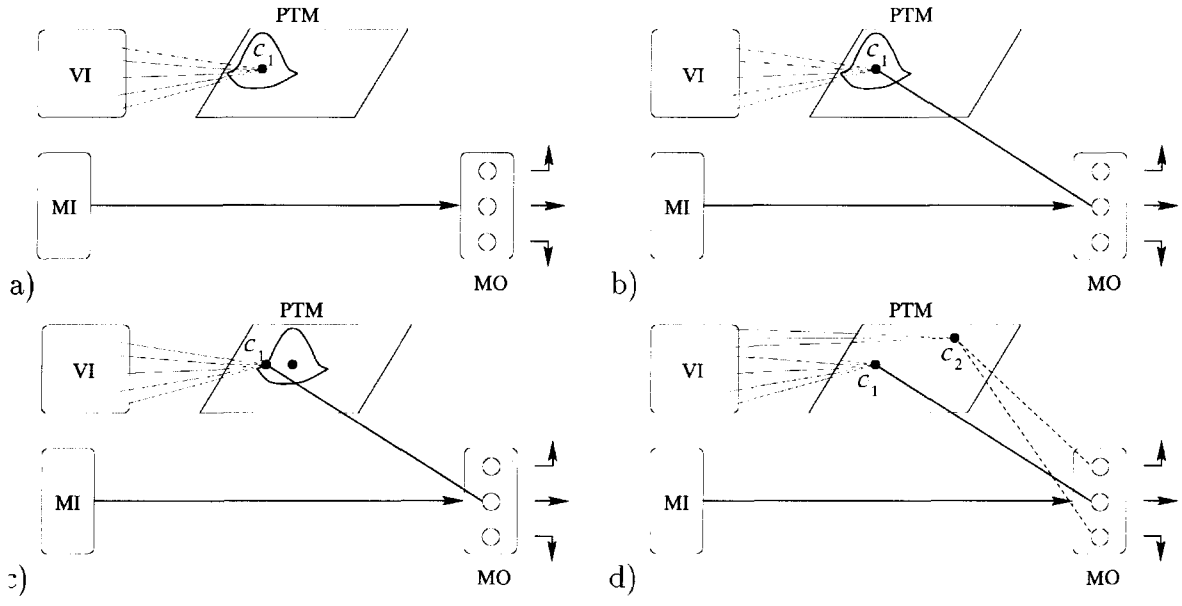


Fig. 11. The activity on the map when the robot moves in the maze: (a) no reaction is learned yet but a first category C_1 is created; (b) after the robot has performed a correct movement, a link between category C_1 and the movement performed is created; (c) a situation similar to (a) makes the robot perform the same movement; (d) if the robot collides with the wall, a new category C_2 is created and can be linked with other movements.

backwards 50 cm and it takes a new image which must look like image e. The vigilance term is then set to 1. As the robot is much more watchful, it perceives that the image is very different from the image corresponding to the pattern coded in C_1 to be matched by C_1. A new category C_2 is thus created (see Fig. 11(d)). Then, the robot tries to associate this category to a movement that keeps it from colliding with the wall. It can now turn left or right and learn the association. The problem of learning which movement is really the one which permits to reach the exit of the maze is solved by the PCR process as described in Section 3.1.

Box 2: The Probabilistic Topological Map (PTM). PTM has been designed to take advantage of the main features of the Carpenter and Grossberg adaptive resonance theory (ART [6] and Kohonen's topological map: ART models are very interesting because they allow unsupervised on-line learning, while Kohonen's maps allow a priori topological generalization. Topological representations are interesting for two main reasons. First they allow to preserve, at least locally, the topology of the input information. Two close stimuli would produce close activity patterns on the map. The map also induces an analog representation of the recognition of a given pattern (as in fuzzy logic). Second, they allow dimension reduction of the input space. Thus, topological maps minimize the needed wiring required for local operations. If there is no topology preservation, as in a WTA, all possible associations must be learned. Furthermore, a neuron that has not yet learned something cannot give any information and cannot take advantage of what has been learned by other neurons. On the contrary, with topological preservation, a neuron in the neighborhood of a given winner will respond accordingly depending on its physical distance to it. It thus gives interesting generalization capabilities [39].

The main characteristics of the PTM algorithm are listed below:

- input patterns and synaptic weights take only binary values;
- a "new" pattern has to be learned immediately (incremental learning);
- a selectivity function is used to allow neurons which have stored a shape to have a higher and sharper response than the other neurons;

- topology on the map is maintained by allowing the coding of the intersection of two learned patterns between the two neurons that codes each of them;
- a vigilance term allows to modulate the generalization level of the map and the decision of coding "new" patterns.

When an input pattern is presented, the most active neuron on the map is chosen as the winner. Due to a diffusion mechanism, neurons in the winner neighborhood are also activated and form an "activity bubble". This mechanism is coupled with a learning process which adapts synaptic weights depending on their distance to the winner. As weights take only binary values, weights of the winners' neighboring neurons are modified to store the input pattern with a probability depending on their distance to it. Thus, when the winner has learned a given pattern, it is a neighbor of that winner that is the most likely to respond when a similar pattern is presented. If they are completely different, they get coded farther away on the map with a maximum distance corresponding to the size of the activity bubble. If two different patterns are presented to the map, their intersection will be coded only between their associated neurons. Indeed, our probabilistic mechanism gives more importance to the synaptic weights which are linked to the intersection of the two shapes than to the other weights.

3.2.2. Study of the dynamical update of the visual categories

In the previous paragraph, we have focused on the robot's ability to create autonomously new categories in order to better respond to its environmental requests. We now want to show that the use of a topological map allows the robot to define recognition frontiers by itself.

Coming back to the example above, in the beginning, before the robot collides with the wall, all the situations are put into the same category C_1 and are thus linked to the same movement; "straight ahead" (see Fig. 12(a) and (a′)). After the collision, a new category C_2 is created at a given distance from C_1. Due to the diffusion mechanism, it automatically creates a boundary in the middle of the distance between C_1 and C_2 (see Fig. 12(b)). If a situation is recognized by a neuron situated on the left-hand side of the boundary, it activates the movement corresponding to the recognition of C_1 (i.e. go straight ahead). By contrast, if the situation is recognized by a neuron on the right-hand side, the robot reacts as if it had been in C_2 (see Fig. 12(b′)).

Now imagine that the robot is put in a situation a again. Then, it would begin by going straight ahead until it recognizes something similar to situation e. Let us now suppose that situation c is closer to e than to a and that if the robot turns left, it collides with the wall (see Figs. 13(c) and (c′)). When the robot arrives in c it wants to perform the movement corresponding to the recognition of C_2, but it cannot because it collides with the wall. As explained above, the vigilance term is increased and thus the robot creates a new category C_3 (see Fig. 13(d)).

Yet there is a problem in learning an association with a different movement from the one learned in e. Indeed, as situation c is closer to e than to a and due to the link between C_2 and the neuron of the WTA that codes the "turn movement" (left or right), this movement always wins. The problem is that PCR only learns to create or destroy a link between the input map winner and the output winner WTA. It does not make any difference because it is not the link between C_3 and the "turn movement" which is involved in the decision but the fact that the C_2 is activated due to the diffusion of C_3 activity. In fact there is no link to suppress. A solution is making another neuron of the WTA win. For that purpose, a first solution consists in adding a very important noise to the WTA neurons output so as to generate random movement choice which might be associated with C_3. We have seen in Section 2.2 that this solution could lead to learning problems. We have preferred an alternative solution which consists in using inhibitory links. In this case, instead of having only one PCR excitatory link per association we have a pair of links: one is excitatory, the other is inhibitory. The learning process for inhibitory links is totally equivalent to excitatory links. As C_3 is activated, it can inhibit the activity of the "turn movement". The next time this movement will not be chosen and another movement, randomly taken, wins. In that case, due to the activation of the positive reinforcement signal, PCR learns to associate C_3 and this new movement.

In fact, the boundary between the "straight ahead" and "turn 90°" behavior, between C_2 and C_3, has thus been shifted (see Fig. 13(d′)). Boundaries can thus be reshaped until the robot reacts properly to the constraints imposed by its environment. This topology can be useful only if a certain continuity of the environment is perceived by

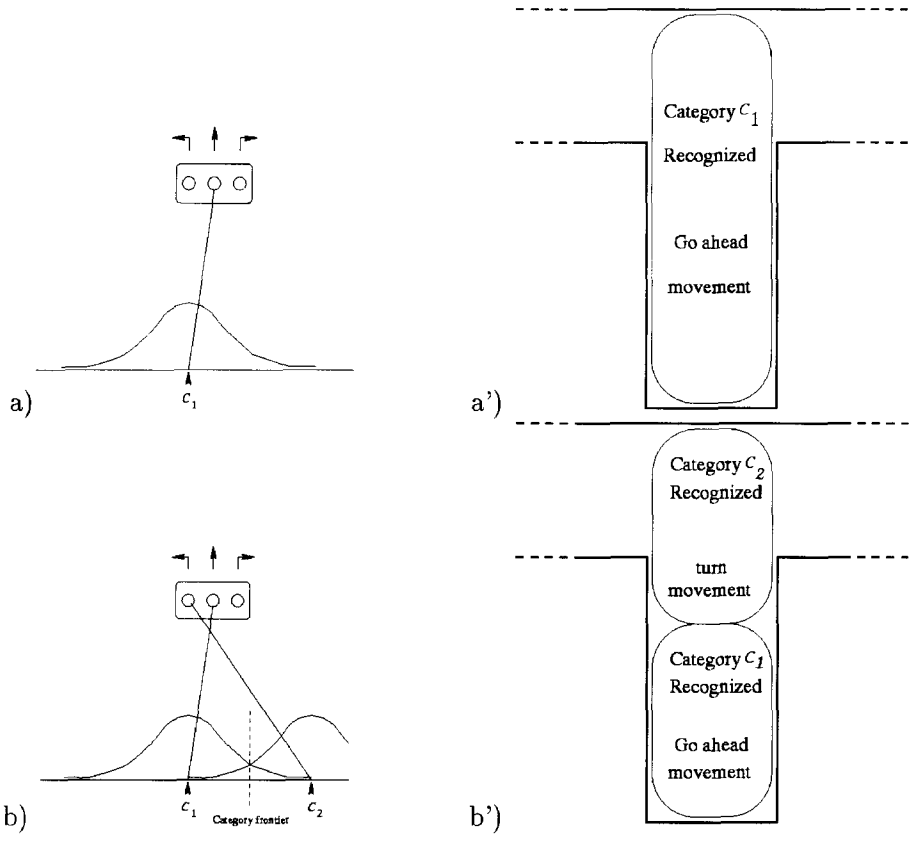


Fig. 12. (a) and (b): Creation of activity bubbles on a one-dimensional map. (a′) and (b′) Corresponding category boundaries in the real maze.

the robot. In fact, the topology is controlled by the vigilance term. If the vigilance is too low, the robot tends to generalize too much and to consider new patterns as patterns already learned. For instance, in a T-junction with a different pictogram from the pictogram it has already seen in another T-junction, it can overgeneralize and react as it would have done if it had been in the other T-junction (even if this movement is no more correct). For another trial in the maze, the vigilance must thus be raised in order for the robot to distinguish the two T-junctions. In conclusion, in order to build a completely autonomous robot, the vigilance term should be self-regulated according to the robot efficiency or to the number of learned shapes (if it learns too many shapes, the vigilance should be reduced).

3.2.3. The reflex system to stay in the middle of the corridor

Our goal is to build a reflex system that enables the robot to have a correct alignment with the corridor. By detecting the edges of the corridor walls, we can determine where they converge. In particular, detecting the vanishing points provides information on the robot position. If there is a vanishing point on the left of the image it is because the robot is too much on the right of the corridor and it should turn left a little. The localization of vanishing points can be obtained by different methods such as a log-polar transform or a Hough transform [27]. Yet, in a neural context, a vanishing point detector can be implemented with orientation sensitive cells. In the simplest case, three vanishing point detection cells are used to sum the activity of orientation sensitive cells distributed in the alignment of the vanishing point (see Fig. 14(a)). For each direction the sum is thresholded so as to suppress non-relevant information. There is one cell in the middle of the image, and two other cells on each side. When a vanishing

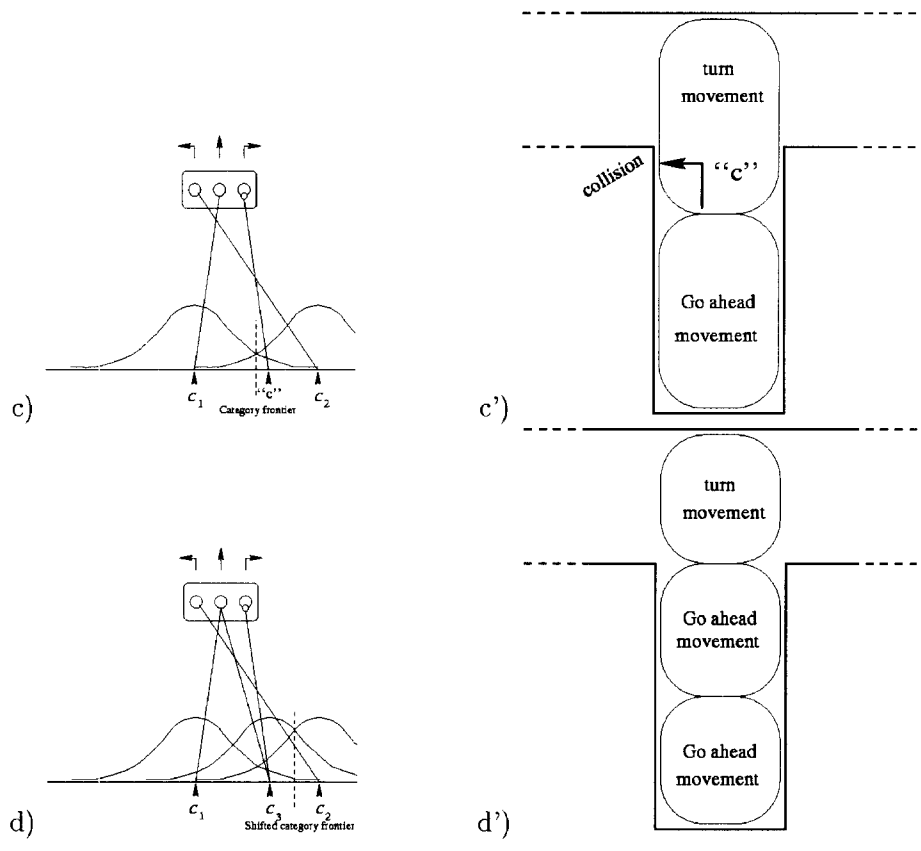


Fig. 13. (c) and (d) Creation of activity bubbles on a one-dimensional map. (c′) and (d′) Corresponding category boundaries in the real maze.

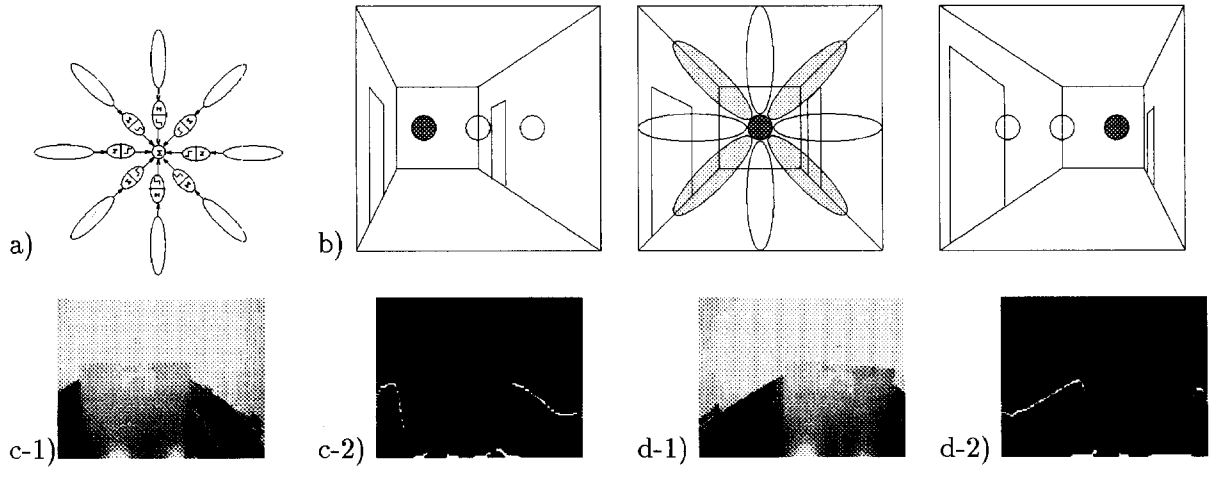


Fig. 14. (a) Vanishing point detection cell. (b) Different kinds of perspectives. (c1) and (c2) Real image and edges: "turn left" situation. (d1) and (d2) Real image and edges: "turn right" situation.

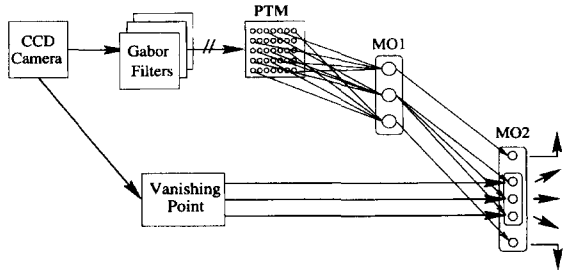


Fig. 15. Complete neural architecture of the robot "brain" used for the maze experiments (the reinforcement system and its input are not represented).

point detection cell reacts it means that the vanishing point is close to this cell (see Fig. 14(b)). The solution of the movement regulation is then to turn slightly to the left when the vanishing point is detected on the right, to turn slightly to the right for the contrary and to go straight ahead otherwise. Other vanishing point detection cells can be added on each side of the image to more precisely locate the position of the vanishing point. The set of vanishing point detection cells is called the movement input group (MI). It is associated by a one to one reflex link to the movement output group (MO) (see Fig. 15).

Figs. 14(c1) and (d1) show examples of the real images the robot sees. The edge extractions (c2) and (d2) allow to take into account edges of the walls which are in the direction of the vanishing point. A threshold on the edge detector is automatically adjusted in a pre-processing phase at the beginning of the experiment in order to adapt to the contrast of the image.

In order to be consistent with the PerAc architecture, the reflex mechanism must directly drive the motor output. Yet, MO1 provides movement information corresponding only to "turn 90°" or "straight ahead" while the MI drives small movements. They do not address the same kind of movements. In fact they belong to two distinct levels: a high level corresponding to the recognition of a specific category and a low level corresponding to the corridor following problem. If the high level has recognized a corridor situation ("go straight ahead"), it triggers the low level movement reflex. Of course, the associations between the vanishing cells and the robot movements to avoid obstacles could have been learned [14]. However, this "reflex learning" must be completed before the beginning of the maze learning in order to overcome combinational explosion (hierarchical learning [23]). The complete architecture of our neural network is displayed in Fig. 15.

3.2.4. Comment on the robot trajectories

In Fig. 16, we present a trajectory of the robot in a simple T-maze. The maze is 3 m wide and 1.8 m long. At the beginning, the robot is put on the right-hand side of the corridor. It can be seen that it succeeds in centering after two or three steps (one step$\simeq$20 cm). As it arrives at the T-junction, it has no difficulty in turn 90° to the left and after this movement it is still centered. It can also be seen that if the movement is not performed correctly (left side of the trajectory), the reflex mechanism enables the robot to re-center itself. Other experiments have been tested in corridors that are slightly curved in a given direction. The robot then manages to follow the curve in order to stay in the middle of the corridor.

In fact this reflex system makes the global architecture very tolerant to slight defaults in orientation or location of the robot. Let us come back again to the example of the robot in a simple T-maze. In Fig. 17, it can be seen that, wherever the robot is in the region B and whatever its orientation (if it respects the orientation given by the reflex system), it can perform a 90° left turn (it is simply a problem of geometrical construction). It relies on the fact that, intrinsically, there is a relationship between the perceived continuity of the images and the continuity of the position of the robot in space. Due to PTM topological properties, this continuity is taken "for free" from the environment. Yet, there might be a problem in region C. In that region, if the robot decides to turn left, it collides with the wall.

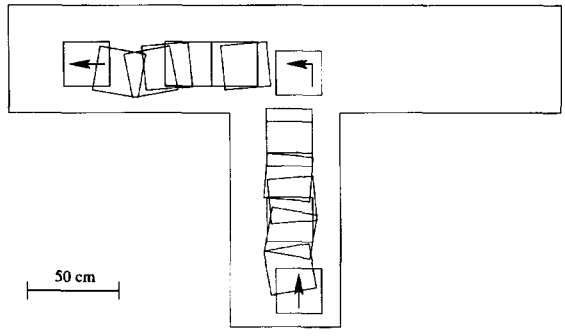


Fig. 16. Record of the robot exact positions in a simple T-maze.

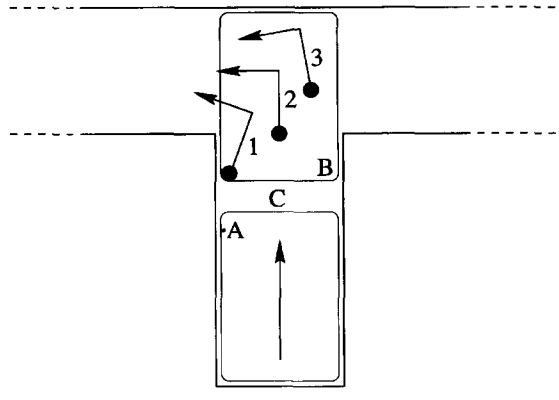


Fig. 17. Tolerance in orientation and position provided by the reflex system. If the robot is in region A (for instance in location 1, 2 or 3), there is no problem to turn left 90° whereas if it is in region C, it collides with the wall.

First of all, it must be noted that, as region C is not very large, the probability to be in that region is low. Besides we have seen previously that PTM was able to create a new category if the robot collided. Thus, regions A and B get larger and larger and the problematic region C must quickly disappear.

We have used PTM topological properties in order to have the same reaction when the robot sees similar images. In fact, it is made possible because of the intrinsic continuity of the environment. This continuity can also be used "for free" because the process of pattern learning respects the continuity too: similar patterns are coded close to each other and a pattern which is intermediate between two other patterns is coded in the middle of the neurons which code these patterns. Yet, this continuity of the environment is only local. After the robot has turned 90°, for instance, the image is totally different from the image it has taken before turning. Therefore, the neuron coding that new image may well be very far from the one coding the previous pattern. The continuity of the map is thus only local.

4. Simulation, tests and discussion

We have shown on a real experiment that PCR algorithm was able to make the robot learn sensory–motor associations that allowed it to reach a given goal. Yet, as this algorithm is stochastic, we need to test it exhaustively in order to show how it statistically converges. But this process is very "time-consuming" and requires important

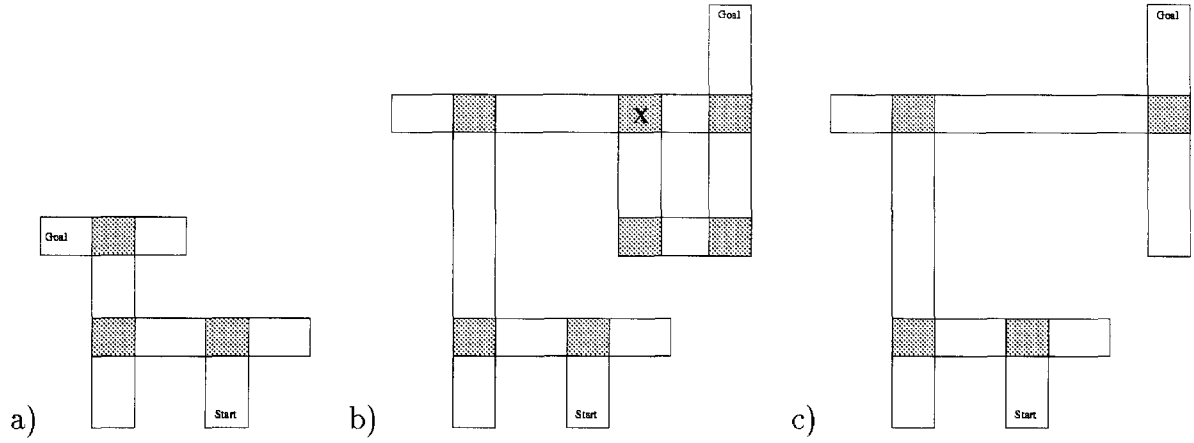


Fig. 18. (a) Short maze with three T-junctions. (b) Long maze with loop. (c) Long maze without loop.

material and human resources. In the first part of this section, we have thus chosen to validate the efficiency of the algorithm on a maze simulation. Yet, we keep the whole architecture and we use images taken by the robot. It allows, in particular, to keep PTM's generalization properties and dynamical state definition.

4.1. Description of the simulation process

Each of the simulated experiments corresponds to the behavior of a single robot. In order to see the influence of the maze size on the learning, three kinds of mazes are used (see Fig. 18). The short maze (Fig. 18(a)) has 30 squares (positions) and the shortest path is 20 squares long (steps). The long maze with a loop (Fig. 18(b)) has 62 squares. The long maze without a loop (Fig. 18(c)) has 49 squares. For both, the shortest path is 40 squares. Moreover, in the simplest case, there are only three patterns, one per direction (all "turn left" T-junctions show the same pattern). In the most complex task (four patterns), each T-junction presents a different pattern (three patterns for the three T-junction + one pattern for "go straight ahead").

4.2. Results and discussion

Results are given as histograms representing 100 robot explorations. Fig. 19(a) represents the number of robots that succeeded in finding the correct set of associations within a given number of trials for the different kinds of mazes. Fig. 19(b) makes the comparison of convergence mean time (in time steps) for the same mazes. As can be seen, the results are very heterogeneous: most of the experiments take only a few number of trials to succeed but some of them require much more time (due to "bad" random draws).

For the mean value, no difference can be seen between learning a "long maze" and a "short maze" for a given complexity (see Fig. 19(a)). This is not surprising because the robot only learns a set of sensory–motor association without regard to their appearance frequency. So, independently of the maze's size, the problem is always to test the efficiency of a behavior during a time long enough to know if it fits the problem well. This is given by a time constant τ_{maze} which must be greater than the shortest time to find the solution (for instance, $\tau_{\text{maze}} = 22$ time steps for the short maze and $\tau_{\text{maze}} = 45$ for the long mazes). It is also important to see that the robot manages to find the solution even if there is a loop inside the maze (as in Fig. 18(b)). Moreover, the loop is of no importance in discovering the solution (the mean numbers of trials are the same for the short and the long maze if their complexities are equivalent – the complexity is defined by the number of associations to be learned). In fact, the mean number of explorations to find the solution depends only on the complexity of the maze.

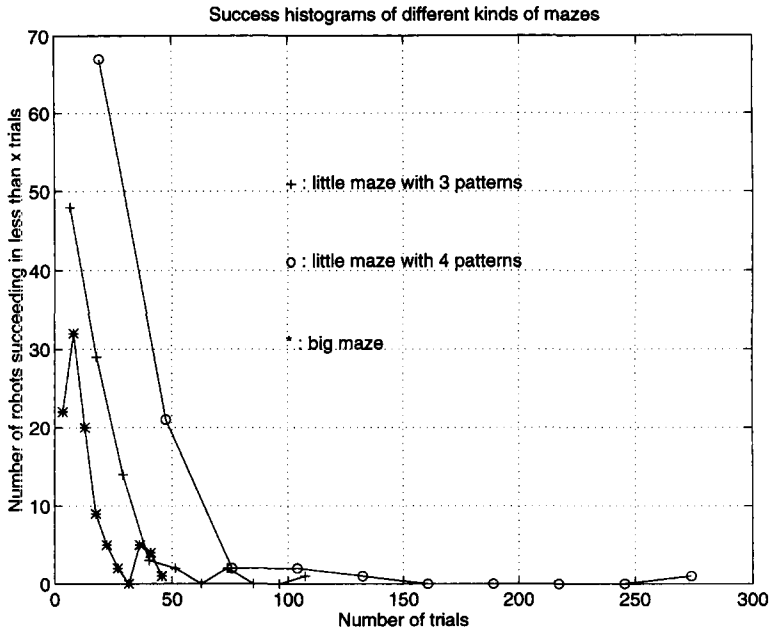


Fig. 19. Histogram of the number of robots which succeed in finding the solution in exactly x trials.

The median value represents the necessary number of explorations for 50% of the robots to succeed in finding the solution. It allows to account for the data heterogeneity. It can be seen that for the case of three patterns, the median and the mean value are very similar. On the contrary, for the case of four patterns, the median is smaller than the mean values, especially for the "large maze" case. This is the indicator of an asymmetry of data distribution that can be explained by the delayed reinforcement signal which forces the robot to find the solution faster. Indeed, in a little maze, this mechanism does not have enough time to work. It thus shows the interest of this mechanism for the convergence speed. Obviously, performances are drastically increased if the reinforcement signal is analogical instead of being binary (i.e. if the reinforcement signal depends on the length of the path). These results are not the concern of this paper because they would involve too long description of the associated reinforcement system.

5. Conclusion and perspectives

In this paper, we have emphasized the need for a global and coherent neural architecture able to control an autonomous robot. We claim that autonomous system designers have to find at once, on-line solutions to both categorization and association problems. We have shown that the PerAc architecture coupled with the probabilistic conditioning rule (PCR) is an efficient approach to solve delayed credit assignments in an indoor environment.

However, in the described NN we have deliberately ignored that some animals can create and use "cognitive maps" of their environment [10,36] to improve their navigation performances. Indeed, if a rat can explore a maze before the beginning of the experiment (without any reinforcement), its learning time to find the shortest path to reach a goal in the same maze will be drastically reduced as soon as it finds the goal (in comparison to "naive" rats). In our current research, we try to fill the gap between models of outdoor navigation [15] and goal seeking and cortical explanations of planning [1]. Then, we hope to have at the same time reinforcement and planning capabilities. Another important

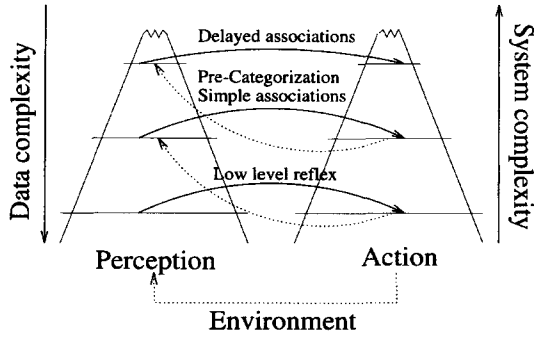


Fig. 20. Representation of the system complexity versus the data complexity in our hierarchical PerAc architecture for complex sensory–motor associations.

issue is to modify the neuron model so as to introduce continuous time considerations. Neuron simulation by differential equations should avoid harsh neuron activation and allow the use of dynamic fields modeling for the control of autonomous robots [33].

In conclusion, it can be noticed that in the artificial life (AL) approach, nowadays, most of the robots use simple and explicit sensors that must directly fit an interpretation (for instance: to detect only the red objects in a scene, you can use a photosensor sensible to red color!). But if the sensors are "omniscient" with regard to the task and the universe, both can be considered as trivial. The main problem is solved by the sensor-designer and not by the roboticists. There is no need for the robot to build categories on its own and the "symbol grounding problem" is not addressed [13,16]. As a result, on a second level of this AL approach, the goal is to build robots capable of working in complex universes. A complex universe can be characterized as a universe without "omniscient" sensors. Its information needs to be analyzed or interpreted before a decision can be taken. For instance, several types of red objects may exist. The system must also build an internal representation of the object to recognize. In that case, the "interesting" objects are supposed to be in the perception field of the robot. Unfortunately, in a real application with an image coming from a CCD camera, the robot has to decide which object in a complex visual scene must be associated to a particular action, so that there will be a combinatorial explosion. The choice must also be restricted to potentially good shapes; i.e. to shapes that can be relevant in robot movement situations. So, the architecture of an intelligent system must reduce, at the same time, the perception and action flow (see Fig. 20).

In this paper, we have only shown how to implement three levels of the architecture depicted in Fig. 20 in a simple case. Indeed, we have deliberately simplified the visual part of the system by making the assumption there is a single object in the scene (Gabor filters are used and designed to approximately match the "interesting" shapes). The conclusion is that we have not solved the following general problem: How can a robot learn to find what can be relevant for its analysis and at the same time perform delayed sensory–motor associations? If a robot or an animal has to face all the possible associations at the same level, it would never be able to find a solution because of the combinatorial explosion. For instance, if we consider a sequence of images there can be more than hundreds of possible objects in each image that could be associated to a particular action with the help of a delayed reward. The limitations of reinforcement mechanisms would make it impossible for an animal or a robot to address very complex universes such as outdoor environments, which is nonsense. That combinatorial problem could be partly solved assuming a Darwinian process and an ontological development of the perception–action system (for a discussion about this problem see [12]).

But these mechanisms cannot explain everything. Indeed, there can be several well-recognized (and learned) objects in a visual scene (the best recognized object might be not relevant for the task). An intelligent system should be able to build a set of hypotheses. Obviously, if the focus of attention mechanism used to select possibly relevant objects chooses an incomplete or biased set of hypotheses, the system will be unable to solve the problem. It is

exactly what seems to happen for children who do badly at school [21]. Thus, the architecture for the control of an autonomous robot that use complex sensors (such as vision) cannot be reduced to simple sensory–motor associations. One cannot hope to scale up behavior based learning strategies to this kind of problems because vision problem involve complex and interdependent internal functionalities. The information provided by the neurobiology and experimental psychology are certainly the only way to mark out the direction in which we must search well-adapted architectures.

Acknowledgements

We would like to thank I. Fijalkow, M. Quoy and R. Pfeifer and reviewers for their helpful comments. We also like to thank the LAMI team at the EPFL and especially F. Mondada, E. Franzi and A. Guignard for designing our prototype of the Koala robot.

Appendix A. The PTM rules

The algorithm for the probabilistic topological map is the following:

(1) Present an input vector I to the map.

(2) Find the winner N^*, i.e. the map cell with highest activity. The similarity between I and the weight vector is first computed and the real neuron activity is processed through an activation function f_k with variable selectivity $\tilde{D}_k(t)$.

The activity in neuron N_k under presentation of binary input vector I is measured as follows:

(3) $Act(N_k, I) = f_k[s_{\text{input}}(W_k, I)] + \text{noise}$.

(4)
$$\begin{cases} \text{if } \tilde{D}_k < 1, \\ \quad s_{\text{input}}(W_k, I) = \dfrac{1}{2} \cdot \left(\dfrac{\sum_{\omega=1}^{P} V_k^{\omega} \cdot I_{\omega}}{S} + \dfrac{\sum_{\omega=1}^{P} \overline{W}_k^{\omega} \cdot \overline{I}_{\omega}}{P - S} \right), \\ \text{otherwise} \\ \quad s_{\text{input}}(W_k, I) = \dfrac{\sum_{\omega=1}^{P} V_k^{\omega} \cdot I_{\omega} + \sum_{\omega=1}^{P} \overline{W}_k^{\omega} \cdot \overline{I}_{\omega}}{\sum_{\omega=1}^{P} V_k^{\omega} + \sum_{\omega=1}^{P} \overline{W}_k^{\omega}}, \end{cases} \tag{A.1}$$

where W_k is N_k's weight vector and

$$\overline{W}_k^{\omega} = 1 - W_k^{\omega}, \qquad \overline{I}_k^{\omega} = 1 - I_k^{\omega}, \qquad V_k^{\omega} = W_k^{\omega} + \alpha \cdot \tilde{W}_k^{\omega}. \tag{A.2}$$

(5)
$$f_k[x](t) = \tilde{D}_k(t) \exp\left[-\frac{\nu + \epsilon}{2\sigma} \left(\frac{1 - x}{1 - \tilde{D}_k(t) + \epsilon} \right)^2 \right]. \tag{A.3}$$

(6) $\tilde{W}_k$ is the vector derived from W_k whose components $\tilde{W}_k^{\omega}$ are equal to 1 when W_k^{ω} has been reinforced more than once and 0 otherwise. P is the dimension of the input space and S is the presumed number of ones of the input vector (it is a constant value).

(7) Diffuse the winner activity on the map according to a diffusion function

$$D_k(t) = D_k(N_k, N^*)$$

(a) $= \exp[-u \cdot d^2_{\text{map}}(N^*, N_k)]$.

(8) If $D_k(t) - \tilde{D}_k(t) > \nu$ (the vigilance parameter) enable learning on the map:

(a) If random $< D_k(t)$ then adapt N_k's weight W_k:

(i) $W_k^{\omega}(t+1) = I^{\omega}$,

(ii) If $I^{\omega}=1$ and $W_k^{\omega}(t)=1$ then $\tilde{W}_k^{\omega}(t+1)=1$,
Else $W_k^{\omega}(t+1) = W_k^{\omega}(t)$.

(b) Modify N_k's selectivity parameter

$$\tilde{D}_k(t+1) = D_k(N^*, N_k).$$

In our experiment the different constants are:

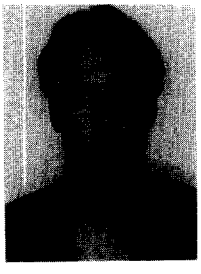
$$\epsilon = 0.5, \quad \sigma = 6, \quad \alpha = 8, \quad P = 15 * 11 * 12 = 1980, \quad S = 150.$$

The vigilance parameter (similar to ART one) allows us to decide whether a new pattern is coded or not. In fact, PTM computes the difference between the maximum activity of all the winner neighbors and their present activity, and compares it with the vigilance term. If it is greater, it is considered that the pattern must be coded. In the other case, the pattern is similar to a pattern already learned, and it is not worth differentiating it.

References

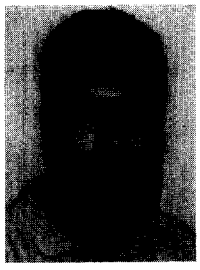
[1] J.S. Albus, Outline for a theory of intelligence, *IEEE Transactions on systems, Man and Cybernetics* 21 (3) (1991) 473–509.
[2] A.G. Barto and R.S. Sutton, Landmark learning: An illustration of associative search, *Biological Cybernetics* 42 (1981) 1–8.
[3] A.G. Barto, R.S. Sutton and C.W. Anderson, Neuronlike adaptive elements that can solve difficult control problems, *IEEE Transactions on System, Man and Cybernetics* SMC-13 (5) (1983) 834–846.
[4] A.G. Barto, R.S. Sutton and D.S. Brouwer, Associative search network: A reinforcement learning associative memory, *Biological cybernetics* 40 (1981) 201–211.
[5] R.A. Brooks, A robust layered control system for a mobile robot, *IEEE Journal of Robotics and Automation* 2 (1) (1986) 14–23.
[6] G.A. Carpenter and S. Grossberg, Invariant pattern recognition and recall by an attentive self-organizing ART architecture in a nonstationary world, *Proceeding of Neural Network* 2 (1987) 737–745.
[7] J.P. Changeux, *Neuronal Man: The Biology of Mind* (Oxford University Press, Oxford, 1985).
[8] R. Chatila, Deliberation and reactivity in autonomous mobile robots, *Robotics and Autonomous System* 16 (2–4) (1995) 197–211.
[9] G. Edelman, *Neural Darwinism: The Theory of Neuronal Group Selection* (Basic Books, New York, 1987).
[10] C.R. Gallistel, *The Organization of Learning* (MIT Press, Cambridge, MA, 1993).
[11] P. Gaussier and J.P. Cocquerez Neural networks for complex scene recognition: Simulation of a visual system with several cortical areas, *IJCNN*, Baltimore (1992) 233–259.
[12] P. Gaussier, C. Joulain and A. Revel, Are shaping techniques the correct answer for the control of an autonomous robot?, in: *UKACC Int. Conf. on Control'96*, University of Exeter (IEE Press, London, 1996).
[13] P. Gaussier and S. Zrehen, Avoiding the world model trap: An acting robot does not need to be so smart!, *Journal of Robotics and Computer-Integrated Manufacturing* 11 (4) (1994) 279–286.
[14] P. Gaussier and S. Zrehen, A topological map for on-line learning: Emergence of obstacle avoidance, in: *From Animals to Animats: SAB'94*, Brighton (MIT Press, Cambridge, MA, 1994) 282–290.
[15] P. Gaussier and S. Zrehen, PerAc: A neural architecture to control artificial animals, *Robotics and Autonomous Systems* 16 (2–4) (1995) 291–320.
[16] S. Harnad, The symbol grounding problem, *Physica D* 42 (1990) 335–346.
[17] R. Hecht-Nielsen, Counterpropagation networks, *Applied Optics* 26 (1987) 4979–4984.
[18] C. Joulain and P. Gaussier, What can robots take for free? Learning to build visual categories from sensori-motor associations, Technical Report, ETIS-ENSEA, 1996.
[19] J. Konorski, *Conditioned Reflexes and Neuron Organisation* (University Press, Cambridge, 1948).
[20] I. Krechevsky, The genesis of "hypotheses" in rats, *Univ. Calif. Publ. Psychol.* 6 (4) (1932) 46.
[21] M. Levine, Hypothesis theory and nonlearning despite ideal S-R-reinforcement contingencies, *Psychological Review* 78 (2) (1971) 130–140.
[22] D.A. Lieberman, *Learning: Behavior and Cognition*, 2nd Ed. (1993).
[23] L.-J. Lin, Programming robots using reinforcement learning and teaching, *Proc. 9th National Conf. on Artificial Intelligence* (1991) 781–786.
[24] M.L. Littman, Memoryless policies: Theoritical limitations and practical results, in: D. Cliff, P. Husbands, J.A.Meyer and S.W.Wilson, eds., *From Animals to Animats: SAB'94* (MIT Press, Cambridge, MA, 1994) 238–245.
[25] S. Mahadevan and J. Connell, Automatic programming of behavior-based robots using reinforcement learning, in: *Proc. 9th National Conf. on Artificial Intelligence*, Menlo Park, CA (1991).

[26] J.L. McClelland, D.E. Rumelhart and G.E. Hinton, The appeal of parallel distributed processing, in: PDP (MIT Press, Cambridge, MA, 1986).
[27] M. Meng and A.C. Kak, Mobile robot navigation using neural networks and nonmetrical environment models, *IEEE Control Systems* (1993) 30–39.
[28] J.R. Millan, Learning efficient reactive behavioral sequences from basic reflexes in a goal-directed autonomous robot, in: D. Cliff, P. Husbands, J.A. Meyer and S.W. Wilson, eds., *From Animals to Animats: SAB'94* (1994) 266–274.
[29] U. Nehmzow and T. Smithers, Mapbuilding using self-organising networks, in: J.A. Meyer and S. Wilson, eds., *From Animals to Animats: SAB'91* (MIT Press, Cambridge, MA, 1991).
[30] R. Pfeifer and C. Scheier, Sensory–motor coordination: The metaphor and beyond, *Robotics and Autonomous Systems* (1996).
[31] R. Pfeifer and P. Verschure, The challenge of autonomous systems: Pitfalls and how to avoid them, in: *The Artificial Life Route to Artificial Intelligence* (MIT Press, Cambridge, MA, 1994).
[32] G.N. Reeke, O. Sporns and G.M. Edelman, Synthetic neural modeling: The "darwin" series of recognition automata, in: C. Lau and B. Widrow, eds., *IEEE Proc., Special issue on Neural Networks* (IEEE Press, New York, 1990) 1498–1530.
[33] G. Schöner, M. Dose and C. Engels, Dynamics of behavior: Theory and applications for autonomous robot architectures, *Robotics and Autonomous System* 16 (2–4) (1995) 213–245.
[34] B.F. Skinner, *Science and Human Behavior* (Macmillan, New York, 1953).
[35] J. Stewart, The implication for understanding high-level cognition of a grounding in elementary adaptive systems, *Robotics and Autonomous Systems* 16 (2–4) (1995) 107–116.
[36] E.C. Tolman, Cognitive maps in rats and men, *Psychological Review* 55 (4) (1948).
[37] T. Trabasso, Stimulus emphasis and all-or-none learning of concept identification, *Journal of Experimental Psychology* 65 (1963) 395–406.
[38] S.E. Weaver, A.H. Klopf and J.S. Morgan, A hierarchical network of control systems that learn: modeling nervous system function during classical and instrumental conditioning, *Adaptive Behavior* 1 (3) (1993) 263–319.
[39] S. Zrehen and P. Gaussier, Why topological maps are useful for learning in an autonomous agent, in: J.D. Nicoud and P. Gaussier, eds., *From Perception to Action Conference*, Los Alamitos, CA (IEEE Press, New York, 1994).

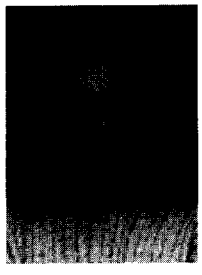


Philippe Gaussier was born in 1967 in Marseille, France. He received the M.S. degree in Electronics from Aix-Marseille University and a Ph.D. degree in Computer Science from the University of Paris XI (Orsay) in 1989 and 1992, respectively. From 1992 to 1994, he conducted research in Neural Network Applications and in control of Autonomous Mobile Robots at the Swiss Federal Institute of Technology. He was the Program Chairman of the Perception Action (PerAc'94) Conference organized in September 1994 in Lausanne. He is now an Assistant Professor at the ENSEA (National Engineering School in Electronics and Its Applications) in France and he leads the group in neurocybernetic. He has worked on various problems related to industrial applications of Neural Networks, visual recognition, robot navigation, neural modeling, etc. Currently his research interests include modelization of visual and navigation systems and their implementation on real robots.

Arnaud Revel was born in 1970 in Angers, France. He is a graduate engineer of the Ecole Nationale Supérieure de l'Electronique et de ses Applications (ENSEA) de Cergy-Pontoise (graduate in computer science). He is currently pursuing a French Ph.D. in the Neurocybernetic team of the image and signal processing lab of the ENSEA. His research interests are to develop neural architectures and learning algorithms inspired by biology and psychology in order to control an animal-like mobile robot.



Cédric Joulain was born in 1972 in Poitiers, France. He received his engineering degree from the Ecole Internationale des Sciences du Traitement de l'Information (EISTI) de Cergy-Pontoise (graduate in computer science) in 1995. He is currently pursuing a French Ph.D. in the Neurocybernetic team of the image and signal processing lab of the Ecole Nationale Supérieure de l'Electronique et de ses Applications (ENSEA) de Cergy-Pontoise. His research interests are to develop neural architectures and learning algorithms inspired by biology and psychology in order to control an animal-like mobile robot.



Stephane Zrehen was born in 1967 in Paris, France. He received his Physics Engineer degree from EPFL in 1990. For two years, he worked as a consultant for industries in Switzerland. Since 1992, he has been working as a research and teaching assistant at the Microcomputing lab at EPFL. He was in charge of the Postgraduate Course on Biological and Artificial Neural Nets. He received his Ph.D. from EPFL in June 1995. His research interests are to keep on developing animal-like intelligent control systems for mobile robots.