**Invited lecture at the 15th Intl. Conf. on Pattern Recognition, (ICPR-2000), vol. 2, pp. 172-177, Barcelona, Sept. 2000.**

# Neuroadaptive Robots

Carme Torras
Institut de Robòtica i Informàtica Industrial (CSIC-UPC)
Gran Capità 2-4 (edifici NEXUS), 08034-Barcelona.
ctorras@iri.upc.es, http://www-iri.upc.es/people/torras

## Abstract

*The limited adaptivity of current robots is preventing their widespread application. However, nowadays there are mature techniques available to palliate this deficiency. After briefly surveying the several levels of adaptivity required and the disciplines addressing each of them, the paper concentrates on the contributions of the field of Neural Networks to improve sensorimotor adaptivity. Since sensorimotor mappings lie at the base of all robot activity, making them adaptable to the robot conditions (e.g., tear-and-wear) and environmental variations greatly widens the range of applications. Several experimental systems are described which rely on the following adaptive mappings: inverse kinematics, inverse dynamics, visuomotor and force-control mappings. Finally, some methodologic and computational issues are discussed.*

## 1. Introduction

Why are robots still confined to factory floors and research departments? Will they ever step out and be part of our everyday lives? Aside from ethical considerations and marketing strategies, there are technological reasons that explain why the use of robots is not as widespread as some envisaged they would be by now. At the risk of oversimplification, let me state that the Achilles heel of current robots is their lack of adaptivity, at all levels. This capability is dispensable in well-engineered environments, and thus we have very performant robots in manufacturing lines, but it is a *sine qua non* when tasks are to be carried out in non-predefined worlds.

In this sense, the biological world –where adaptivity is crucial for survival– constitutes a very good source of inspiration for robotics researchers, since it provides existence proofs of many adaptive mechanisms that do function. However, caution must be taken, because the best natural solution may not be the best artificial one [24]. Wheels, wings and calculators have often been mentioned as examples of artificial solutions considerably different from their natural counterparts, and more performant according to certain criteria. The resources available to engineering design depart a lot from those in nature, and not just when it comes to materials, but also in the number of instances and spendable time.

With this note of caution in mind, i.e., accepting that biological plausibility in itself adds no special value from an engineering viewpoint, it is safe to look into natural adaptivity to get seed ideas that can be instantiated in a different way by artificial means.

## 2. Natural and artificial adaptivity

What exactly do we mean by adaptivity? What does it encompass? What is its range? By adaptivity we mean the capability of self-modification that some agents have, which allows them to maintain a level of performance when facing environmental changes, or to improve it when confronted repeatedly with the same situation. The term 'agent' above stands for a single cell, an organ, an individual or even a whole society, because, in the biological world, adaptivity occurs at several levels, each having a possible counterpart in the design of autonomous robots [25].

At the cell level, several chemical and electrical mechanisms of **plasticity** have been discovered, some of which have been modelled and analysed within the Neural Networks field [2], and later applied to adjust the parameters of robot sensors and actuators.

At the sensorimotor level, adaptation takes the form of an **association**, built through either classical or instrumental conditioning, as studied within the Behavioural Psychology field. Again, neural network models able to build relevant associations from experience [7, 27] have been applied to the construction of robot sensorimotor mappings. This is the central topic of this paper and will be discussed in the following sections.

At a cognitive level, several **symbolic learning** strategies have been postulated, some of which have been mimicked within the field of Artificial Intelligence and later incorporated into learning robots [15, 17].

Finally, at the species level, adaptation is attained through **evolution**. Genetic algorithms [11] and evolutionary computation [6] are starting to be used to tailor robot genotypes to given tasks and environments [8].

## 3. Neural adaptivity

Neural Networks are essentially procedures for approximating nonlinear mappings given a set of inputs and some information on the corresponding outputs. The approximation is attained by iterative tuning of the connection weights. Depending on the amount of output information required, three types of procedures can be distinguished: correlational, reinforcement and error-minimization procedures (Figure 1).

**Correlational procedures** use no output information and their goal is to carry out feature discovery or clustering in the input space. The most widely used procedures of this type are *self-organizing feature maps (SOM)* [10], *the cerebellar model articulation controller (CMAC)* [1] and *adaptive resonance theory* [4]. These procedures are often used to represent a given state space in a compact and topology-preserving manner. Two systems to be described later rely on representations of this type for the robot workspace (see Section 4.1) and the space of joint positions, velocities and accelerations (see Section 4.2).

**Error-minimization procedures** require complete target information –in the form of input/output pairs– and their goal is to build a mapping from inputs to outputs that generalizes properly. Two such procedures, namely the *LMS rule* [32] and *back-propagation* [13, 21], have been the most widely used in applications. Since back-propagation has the drawbacks of all gradient descent techniques, namely the possibility of getting stuck in local minima and a slow convergence rate, numerous acceleration procedures have been proposed. Moreover, back-propagation suffers from catastrophic forgetting of the previously learnt patterns when trained with a new pattern. Thus, techniques to prevent forgetting by explicitly minimizing degradation while encoding a new pattern have been devised [19]. Finally, as we will see later, some authors have also used *conjugate gradient* optimization techniques.

**Reinforcement-based procedures** lie between both extremes. They make use only of a reward/penalty signal to build a mapping that maximizes reward. As trial-and-error procedures, they have a stochastic component that permits exploring the suitability of different outputs in response to the same input. The *associative search learning rule* [3], the *method of temporal differences* [26] and *Q-learning* [9] are the most well-known procedures of this type.

## 4. Robot sensorimotor mappings

Robot tasks are usually specified in world coordinates (or, alternatively, in terms of sensor readings), while robot moves are governed by their actuator's variables. Therefore, robot control critically depends on the availability of accurate mappings from physical space or sensor space to joint space or motor space. The discussion in what follows is centered on mappings required for arm robots to work, but similar arguments apply to the case of mobile robots.

For a robot arm to carry out point-to-point motions, its controller must access an *inverse kinematics mapping*, i.e., that providing joint coordinates as a function of the position and orientation of the robot end-effector. If a desired end-effector trajectory is specified instead, then the controller should resort to an *inverse dynamics mapping* relating such trajectory to the forces and torques that need to be exerted at the different joints to realize it. For tasks entailing the achievement of a goal using sensory feedback, an appropriate *sensorimotor mapping* relating sensory patterns to motor commands is needed.

The aforementioned mappings vary considerably, depending not only on the nature of the involved sensors and actuators, but also on the goal pursued. This diversity sometimes hides what they have in common: an underlying highly nonlinear relation between a continuous (often hard to interpret) input domain and a continuous motor domain; a relation that is very difficult (when not impossible) to derive analytically. Furthermore, because of environmental changes or robot tear-and-wear, the mappings may vary in time and then one would like the controller to adapt to these variations, without any human intervention if possible. Therefore, a way of learning (or tuning) these mappings automatically while robots move is highly desirable. Since, as we have mentioned, neural networks are essentially procedures for approximating nonlinear mappings, they constitute a promising tool to attain the desired adaptivity.

### 4.1. Inverse kinematics

The use of neural networks to approximate the inverse kinematics of robot arms is of particular interest when a precise model of some joints is lacking or when, due to the operation conditions of the robot (in space, underwater, etc.), it is hardly possible to recalibrate it.

Feedforward networks using back-propagation have been extensively tested in this context, leading to the conclusion that a coarse mapping can be obtained quickly, but an accurate representation of the true mapping is often not feasible or extremely difficult. The reason for this seems to be the *global* effect that every connection weight has on the final approximation obtained [12].

A way to avoid this global effect is using local representations, so that every part of the network is responsible for a small subspace of the total input space. For instance, a 3D SOM has been used to encode the robot workspace [18]. This is combined with the LMS rule to learn the inverse kinematics of a robot arm with three degrees-of-freedom (dof). The inputs to each neuron are the coordinates of the desired end-effector

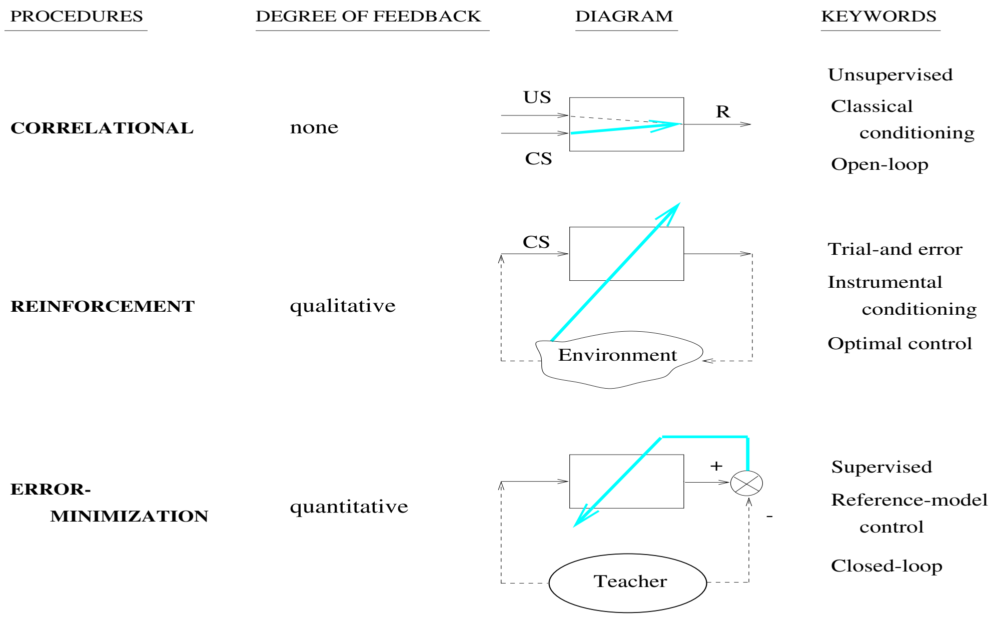


**Figure 1. Procedures for neural adaptivity. See [27] for a detailed explanation.**

position, and the outputs (after correct learning) are the joint angles and the Jacobian corresponding to that position. Thus, this model provides a discrete encoding of the inverse kinematics mapping augmented with a linear approximation at each sample point that permits interpolating joint angles with higher precision. The network has been shown to self-organize into a reasonable representation of the workspace in about 30.000 learning cycles. This should be taken as an experimental demonstration of the powerful learning capabilities of this model, because the conditions in which it was made to operate are the worst possible ones: no a priori knowledge of the robot kinematics, random weight initialization, and random sampling of the workspace during training.

This basic model has been extended in three directions to cope with higher-dof robots. First, a hierarchical version, consisting of a 3D SOM whose nodes have associated a 2D SOM each, was applied to a 5-dof robot. The 3D net encodes the workspace as before, while each 2D subnet approximates the end-effector orientation space at the corresponding position [18].

This hierarchical model has been adapted to suit a practical setting [20]. Thus, instead of learning the kinematics from scratch, only the deviations from the nominal kinematics embedded in the original robot controller are learnt. This, together with informed initialization and sampling, as well as several modifications in the learning algorithm aimed at improving the cooperation between neurons, lead to a speed-up of two orders of magnitude with respect to the original model. Thus, when applied to the self-calibration of a 6-dof robot installed in a space-station mock-up, 95% of the decalibration was corrected with the first 25 movements, this percentage raising to 98% after 100 movements. Moreover, other desirable features in stand-alone applications, such as parameter stability, are guaranteed.

The third extension relies on the generalization of SOMs to Parametrized SOMs (called PSOMs). The idea is to turn the discrete representation into a continuous one by associating a basis function to each neuron, so that a parametrized mapping manifold is obtained. Moreover, PSOMs make no distinction between inputs and outputs, thus encoding bidirectional mappings. The PSOM reduces considerably the number of training samples required to attain a given precision as compared to the SOM [29], allowing the learning of the full inverse kinematics of a 6-dof robot with less than 800 movements.

## 4.2. Inverse dynamics

When the robot dynamics needs to be taken into account, as in trajectory following, the control learning problem becomes more involved. First and second derivatives of both end-effector positions and joint angles need now be considered and, moreover, some sort of fixed robot controller is required to generate the training data.

Since the cerebellum is known to be involved in the production and learning of smooth movements, several cerebellar models have been proposed and applied to control robot arms. The pioneer such model was CMAC, but today the debate is still open as to what model best captures the functionality of the cerebellum and whether any such model can constitute a practical option to control robots [28]. A point of agreement is that the cerebellum constructs an inverse dynamics model as it learns. Thus, cerebellar models have been used for this purpose inside robot controllers.

Miller et al. [14] have combined the table look-up facilities provided by CMAC with an error-correction scheme similar to the LMS rule to accomplish the dynamic control of a 5-dof robot. The idea underlying this combination is similar to that of enlarging SOMs with the LMS rule, as described in the preceding section. Here, CMAC is used to represent the state space in a compact and localized manner, as there SOMs were used to cover the robot workspace. To teach the robot to follow a given trajectory, successive points along it are supplied to both the neural network and a fixed-gain controller and then their responses are added up to command the robot. Therefore, the neural network acts as a feedforward component. After each cycle, the actual command given to the robot together with its current state are used as an input-output pair to train the neural network. As learning progresses, the CMAC network approximates the inverse dynamics mapping and, consequently, the effect of the fixed-gain controller tends to zero. The network converges to a low error (between 1 and 2 position encoder units) within 10 trials, provided enough weight vectors are used.

The same trajectory learning task above has been tackled by Miyamoto et al. [16] by using the so called feedback-error learning approach. They use a 3-dof robot and a neural network with only 3 neurons (one per joint) whose inputs are 13 nonlinear functions of the joint velocities and accelerations. Therefore, 39 weights undergo adaptation using an error-correction rule similar to the LMS one. The teaching scheme is the same used by Miller et al., the two approaches differing in the error signal used to modify the weights. Miyamoto et al. do not generate input-output pairs, but use directly as error signal the output of the feedback controller, which somehow measures the deviation of the current state from the desired one in terms of the control signal required to make the former approach the latter. This error measure is less accurate than that used by Miller et al., but has the advantage of being directly available in the control loop, thus avoiding the computation of the current state of the robot. The authors report that, after training the robot to follow a trajectory lasting 6 seconds for 300 trials, the average feedback torque decreased from a few hundreds to just a few units, demonstrating that the neural network had taken over control from the fixed-gain controller. Moreover, the mean square error in the joint angles decreased steadily 1.5 orders of magnitude.

## 4.3. Force-motor mappings

The possibility of using neural networks to learn the action to apply in response to each force pattern (i.e., the appropriate sensorimotor mapping) looks very attractive.

Gullapalli et al. [5] have used an associative reinforcement learning system to learn active compliant control for peg-in-hole insertion using a 6-dof robot. The system takes the sensed peg positions and forces, as well as the previous position command, as inputs, and produces a new position command as output. Thus, eighteen real values are entered into a network with two hidden layers of back-propagation units, and six real values are produced by its output layer of stochastic reinforcement-learning units. The reinforcement signal depends on the discrepancy between the sensed and the desired position of the peg, with a penalty term being activated whenever the sensed forces on the peg exceed a preset maximum. The training runs start with the peg at a random position and orientation with respect to the hole, and end when either the peg is successfully inserted or 100 time steps have elapsed. Experimental results show that, after 150 trials, the robot is consistently able to complete the insertion. Moreover, the time to insertion decreases continuously from 100 to 45 time steps over the subsequent 500 training runs.

## 4.4. Visuomotor mappings

Depending on the task to be performed and the camera-robot arrangement, visuomotor mappings take different forms. Thus, in eye-hand coordination, where cameras external to the robot are used to monitor the pose (position and orientation) of its end-effector, a mapping from the camera coordinates of a desired end-effector pose to the joint angles that permit attaining that pose is sought. This mapping is closely related to the inverse kinematics one, especially if the camera coordinates of selected points in the end-effector uniquely characterize its pose. Therefore, the same models used to learn inverse kinematics have been applied to the learning of the visuomotor mapping underlying eye-hand cooordination [18].

A camera mounted on a robot arm is used in tasks such as visual positioning and object tracking. The goal of these tasks is to move the camera so that the image captured matches a given reference pattern. The target is thus no longer a position in space but a desired image pattern, and the desired visuomotor mapping needs to relate offsets w.r.t. that pattern with appropriate movements to cancel them. In visual positioning, the scene is assumed to be static and the main issue is to attain high precision. Applications include inspection and grasping of parts that cannot be precisely placed (e.g., in underwater or space settings). The aim of object tracking is to maintain a moving object within the field of view, speed being here the critical parameter instead of precision.

The classical way of tackling these tasks consists of defining a set of image features and then deriving an interaction matrix relating 2D shifts of these features in the image to 3D movements of the camera [22].

In the case of visual positioning, back-propagation has been used to learn the interaction matrix [31]. The training procedure consists of moving the camera from the reference position to random positions and then using the displacement in image features together with the motion performed as input-output pairs. In operation, the robot is commanded to execute the inverse of the motion that the network has associated to the given input. The key option in this work is the use of global image descriptors, which permits avoiding the costly matching of local geometric features in the current and reference images. By using a statistical measure of variable interdependence (the mutual information criterion), sets of global descriptors as variant as possible with each robot dof are selected from a battery of features, including geometric moments, eigenvectors, pose-image covariance vectors and local feature analysis vectors [30]. The results obtained with a 6-dof show that, after 10.000 learning cycles, translation and rotation errors are lower than $2mm$ and 0.1 degrees, respectively.

Concerning object tracking, Schram et al. [23] have used a feedforward network together with a conjugate gradient learning algorithm to make a camera track a cart moving arbitrarily on a table. A visuomotor mapping relating the current and past visual coordinates of the cart with joint displacements is built on-line as the robot moves. Only two robot dofs need to be controlled, and thus the network has two outputs, while several numbers of inputs have been tried. The tracking performance is shown to improve as more previous positions of both the cart and the robot are used, attaining an average lag of only $8mm$ in the case of seven inputs.

## 5. Conclusions

After surveying some experimental results obtained with neuroadaptive robots (Table 1), we conclude that:

* In the case of *mappings that can be easily sampled*, it seems sufficient to apply a plain error-minimization procedure. Some simple inverse kinematics mappings and visuomotor mappings used for visual positioning have been learned in this way.

* If the *input space is complex*, then many researchers have resorted to a combination of correlational rules for the efficient coding of that space, with error-minimization rules to build the appropriate association with the outputs. The use of SOMs to encode the robot workspace or the sensor space, as well as the application of CMAC to the coding of the robot dynamics state space, fall into this category. In both cases, the LMS rule is used to build the appropriate input-output mapping: inverse kinematics in the former case and inverse dynamics in the latter one.

* In the case that a *measure of the error is directly available in the control loop*, as it happens in some inverse dynamics applications, then it seems natural to apply a feedback-error learning approach. The very nature of this approach points towards the use of error-minimization rules.

* Finally, when *the task is specified as a goal to be reached* using sensory feedback, without making explicit the movements necessary to reach it, then the only possibility is to resort to reinforcement learning schemes, which depend just on the availability of a measure of success rather than an error measure.

The number of learning cycles required ranges widely in the applications described, depending on the complexity of the mapping to be learned as well as on the accuracy required. Only 10 trials are needed to get a useful mapping in the case of inverse dynamics using CMAC. The explanation is that only a very coarse mapping is needed, since the neural controller is used as a feedforward component in combination with a fixed-gain feedback controller. The number of trials raises to a few hundreds in the case of force-motor mappings for insertion of components. 100 learning cycles suffice to correct the distortions in the inverse kinematics mapping resulting from robot wear-and-tear, while this number raises to near 1.000 when the full mapping has to be learned from scratch. And the progression continues to up to 10.000 trials when the inputs are not spatial coordinates but global descriptors extracted from images. Of course, some of these figures might be considerably lowered in the future; especially the last one if more efficient codings of the input space are found.

## References


[1] J.S. Albus. A new approach to manipulator control: The cerebellar model articulation controller (CMAC). *Transactions of the ASME, Journal of Dynamic Systems, Measurement and Control*, 97:220–227, 1975.

[2] M.A. Arbib. *Handbook of Brain Theory and Neural Networks*. MIT Press, Cambridge, MA, 1995.

[3] A.G. Barto, R.S. Sutton and P.S. Brouwer. Associative Search Network: A reinforcement learning associative memory. *Biological Cybernetics*, 40:201–211, 1981.

[4] S. Grossberg. Competitive learning: from interactive activation to adaptive resonance. *Cognitive Science*, 11:23–63, 1987.

[5] V. Gullapalli, A.G. Barto, and R. Grupen. Learning admittance mappings for force-guided assembly. *Proc. IEEE Intl. Conf. on Robotics and Automation*, IEEE Computer Society Press, Los Alamitos, CA, 2633–2638, 1994.

[6] T. Higuchi, M. Iwata and W. Liu (Eds.). *Evolvable Systems: From Biology to Hardware*. Springer-Verlag, Berlin Heidelberg New York, 1997.

[7] G.E. Hinton. Connectionist learning procedures. *Artificial Intelligence*, 40:185–234, 1989.

[8] P. Husbands and J.A. Meyer. *Proc. 1st European Workshop on Evolutionary Robotics (EvoRobot'98)*. Springer-Verlag, Berlin Heidelberg New York, 1998.

| MAPPING | CORRELATIONAL + ERROR-MINIMIZATION | ERROR-MINIMIZATION | REINFORCEMENT LEARNING |
|---|---|---|---|
| INVERSE KINEMATICS | **SOM+LMS** [Ritter et al. 92, Ruiz de Angulo & Torras 97]<br>**PSOM+LMS** [Walter & Ritter 96] | **BP** [Jordan & Rumelhart 92, Kröse & van der Smagt 93] | |
| INVERSE DYNAMICS | **CMAC+LMS** [Miller et al. 90] | **LMS** [Miyamoto et al. 88] | |
| FORCE-MOTOR | | | **RL** [Gullapalli et al. 92] |
| VISUOMOTOR | **SOM+LMS** [Ritter et al. 92] | **BP** [Wells et al. 96]<br>**CG** [Schram et al. 96] | |

**Table 1. Neuroadaptive procedures used to approximate several robot mappings**


[9] L. Kaelbling. *Learning in Embedded Systems*. MIT Press, Cambridge, MA, 1993.

[10] T. Kohonen. *Self-Organization and Associative Memory* (second edition). Springer-Verlag, Berlin Heidelberg New-York Tokyo, 1988.

[11] J.R. Koza. *Genetic Programming: On Programming Computers by means of Natural Selection*. MIT Press, Cambridge, MA, 1992.

[12] B.J.A. Kröse and P.P. van der Smagt. *An Introduction to Neural Networks* (5th edition). University of Amsterdam, 1993.

[13] Y. LeCun. Une procedure d'aprentissage pour reseau au seuil assymetrique. *Proceedings of COGNITIVA*, 599–604, 1985.

[14] W.T. Miller, R.P. Hewes, F.H. Glanz and L.G. Kraft. Real-time dynamic control of an industrial manipulator using a neural-network-based learning controller. *IEEE Trans. on Robotics and Automation*, 6(1):1–9, 1990.

[15] T. Mitchell, J. Franklin and S. Thrun. *Recent Advances in Robot Learning*. Kluwer Academic Publishers, Boston, MA, 1996.

[16] H. Miyamoto, M. Kawato, T. Setoyama and R. Suzuki. Feedback-error-learning neural network for trajectory control of a robotic manipulator. *Neural Networks*, 1:251-265, 1988.

[17] K. Morik, M. Kaiser and V. Klingspor. *Making Robots Smarter: Combining Sensing and Action through Robot Learning*. Kluwer Academic Publishers, Boston, MA, 1999.

[18] H. Ritter, T. Martinetz, and K. Schulten. *Neural Computation and Self-Organizing Maps*. Addison Wesley, New York, 1992.

[19] V. Ruiz de Angulo and C. Torras. On-line learning with minimum degradation in feedforward networks. *IEEE Trans. on Neural Networks*, 6(3):657–668, May 1995.

[20] V. Ruiz de Angulo and C. Torras. Self-calibration of a space robot. *IEEE Trans. on Neural Networks*, 8(4):951-963, 1997.

[21] D.E. Rumelhart, G.E. Hinton, and R.J. Williams. Learning representations by back-propagating errors. *Letters to Nature*, 323:533–535, 1986.

[22] C. Samson, M. LeBorgne and B. Espiau. *Robot Control: The Task Function Approach*. Oxford Engineering Science Series 22, Oxford Science Publications, 1990.

[23] G. Schram, F.X. van der Linden, B.J.A. Kröse and F.C.A. Groen. Visual tracking of moving objects using a neural network controller. *Robotics and Autonomous Systems*, 18:293-299, 1996.

[24] H.A. Simon. *The Sciences of the Artificial*. MIT Press, Cambridge, MA, 1969.

[25] L. Steels. *The Biology and Technology of Intelligent Autonomous Agents*. NATO ASI Series F, Springer-Verlag, Berlin Heidelberg New York, 1995.

[26] R.S. Sutton. Learning to predict by the methods of temporal differences. *Machine Learning*, 3:9–44, 1988.

[27] C. Torras. Robot adaptivity. *Robotics and Autonomous Systems*, 15(1-2):11-23, 1995.

[28] P. van del Smagt and D. Bullock (Eds.). Can Artificial Cerebellar Models Compete to Control Robots?. DLR Technical Report #515-97-28, German Aerospace Center, 1997.

[29] J. Walter and H. Ritter. Rapid learning with parametrized self-organizing maps. *Neurocomputing*, 12:131-153, 1996.

[30] G. Wells, and C. Torras. Selection of image features for robot positioning using mutual information. *Proc. IEEE Intl. Conf. on Robotics and Automation*, IEEE Computer Society Press, Los Alamitos, CA, 2819-2826, 1998.

[31] G. Wells, Ch. Venaille and C. Torras. Vision-based robot positioning using neural networks. *Image and Vision Computing*, 14:715-732, 1996.

[32] B. Widrow and M.E. Hoff. Adaptative switching capatibility and its relation to the mechanisms of association. *Kybernetik*, 12:204–215, 1960.