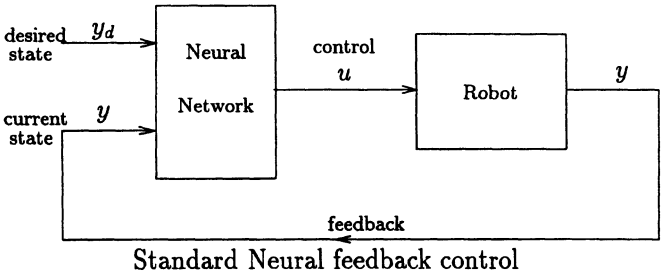
USING BACKPROPAGATION ALGORITHM FOR NEURAL ADAPTIVE CONTROL: EXPERIMENTAL VALIDATION ON AN INDUSTRIAL MOBILE ROBOT

P. Henaff and S. Delaplace
University of Paris 6, Vélizy, France
and
Versailles Saint-Quentin University, Vélizy

Abstract : *This paper presents an original method in the use of neural networks and backpropagation algorithm to learn control of robotics systems. The originality consists to express the control objective as a criterion of which the gradient is backpropagating through the network instead of the classical quadratic error used in standard backpropagation. This technic allows on-line learning that is impossible to do with standard backpropagation. Experimental validation is realised by the position and the orientation control of a faster industrial mobile robot. Results show the feasability of the method, and particularly establish that on-line learning scheme permit to refine the weights of the network in front of the kinematics constraints of the robot.*

1 Introduction : problems of direct backpropagation

Multi-layered networks are very interesting in control of robotic systems because of their known generalization capabilities. A standard neural control scheme is build on the academic feedback control approach where the neural net replace the classical corrector.



Standard Neural feedback control

On right figure, net input is a combination between the current state of the system and a desired state. Net outputs are the control parameters (joint torques for example).

Multi-layered networks are usually trained with the well known backpropagation learning algorithm we named direct backpropagation. The goal of direct backpropagation is to minimize the quadratic error E (addition of quadratic errors E_p for every input pattern p) between the net and a desired output (see Fig. 1). Consequently, it requires an a priori knowledge of the desired outputs (e.g. the desired controls parameters) corresponding to the net input.

A. Morecki et al. (eds.), *ROMANSY 11*
© Springer-Verlag Wien 1997

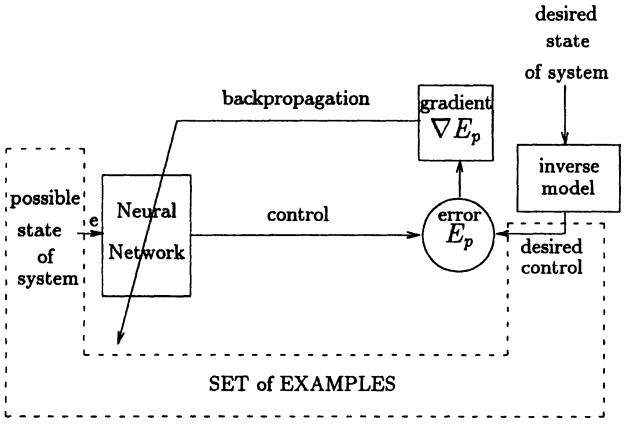


Figure 1: Standard Backpropagation

Determination of desired outputs is the main problem of using standard backpropagation in learning control of robotic applications, because either a reference model is use (PID, fuzzy controller [5], human teacher [2],...) and it does not exploit the whole learning capabilities of neural nets (except for the human teacher); or a inverse model of the robot is use, and the inversion can not, in general, easily be solved on-line.

In the case of non holonomic mobile robots (independent driving wheeled robots) the inverse model can not easily be obtained because of the non-integrability of the equations. Then it is impossible to learn on-line with Direct Backpropagation.

To make allowance for these remarks and since backpropagation is an optimization algorithm, we propose a learning method that doesn't need desired outputs, but only a criterion specifying the robotics task.

The first idea of this approach has been introduced before in [6] for a simulated arm cartesian control problem. Our method has been presented in [8] for a dynamic reflex control problem in which the network learns to control the dynamic equilibrium of a simulated planar biped. In [10], we have presented experimentations on a slow mobile robot that show the feasibility of our learning approach.

In this paper, we presents experimental results on this approach (we named it indirect backpropagation) applied to the control of a faster industrial mobile robot where dynamic effects are very important. Next section describes indirect backpropagation.

2 Solution : learning with indirect backpropagation

The basic idea is to minimize the arbitrary criterion J_p (for every input pattern p) as a function of the net output, instead of the error E_p between the net and the desired output. This criterion explain mathematically the control objective. The advantage of learning with criterion is that we need not any desired output, but only a cost function which specify the control objective and its constraints. So, the neural network determines itself the way to achieve this objective.

In order to minimize this criterion, we use also a gradient descent method :

$$\frac{\partial J_p}{\partial w_{ji}} = \frac{\partial J_p}{\partial net_{pj}} \frac{\partial net_{pj}}{\partial w_{ji}} = \frac{\partial J_p}{\partial net_{pj}} o_{pi}$$

where w_{ji} is the weight from neuron i to neuron j, o_p is the net output vector, $net_{pj} = \sum_i w_{ji} o_{pi}$ denotes the activation of neuron j and f_j its function (sigmoïde).
We set $\delta_{pj} = -\frac{\partial J_p}{\partial net_{pj}}$ and obtain:

- for a hidden neuron : $\delta_{pj}^{hid} = f_j'(net_{pj}) \sum_k \delta_{pk} w_{kj}$
- for an output one : $\delta_{pj}^{out} = -\frac{\partial J_p}{\partial o_{pj}} \frac{\partial o_{pj}}{\partial net_{pj}} = -\frac{\partial J_p}{\partial o_{pj}} f_j'(net_{pj}) = \nabla J\, f_j'(net_{pj})$

Instead of the error vector E_p, we backpropagate the gradient vector of the criterion $\nabla J_p = -\frac{\partial J_p}{\partial o_{pj}}$ related to the corresponding output. Therefore, the partial derivates of the criterion must be analytically calculable with respect to the control parameters. The computation of δ_{pj}^{hid} remains unchanged.

2.1 Learning Off-line and on-line with indirect backpropagation

Figures 2 and 3 illustrate the new learning algorithm (compares with figure 1). It can be use to train the net off-line with a model of the robot and on-line with the real robot.

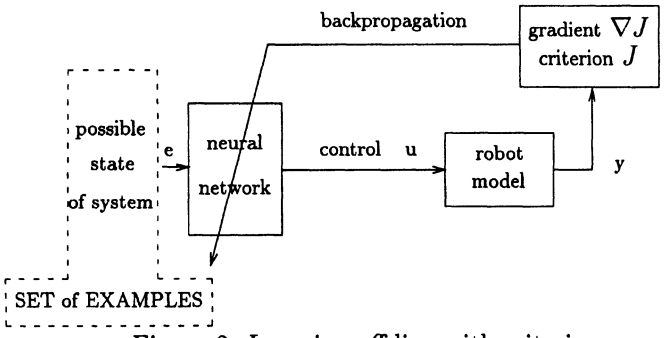


Figure 2: Learning off-line with criterion

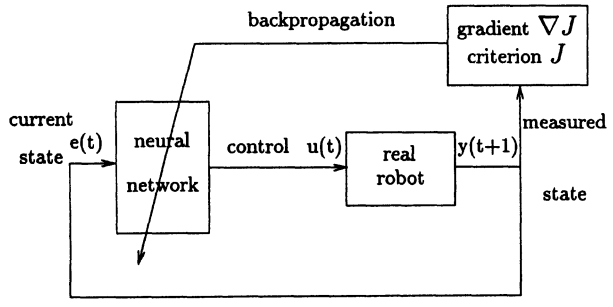


Figure 3: Learning on-line with criterion

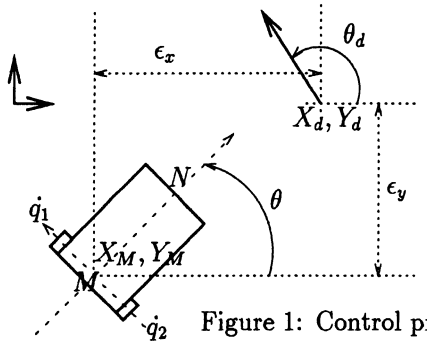
Our learning structure is very simple. The on-line structure allows to refine the neural controller in front of real behavior of the robot. In comparison, D. H. Nguyen and B. Widrow proposed in [3] to use a network to emulate the system. Then, they can backpropagate a criterion through the emulator net and trained the controller without a gradient. Nevertheless, their method allows not to control the robot during the learning because the criterion cannot be propagated through it. Consequently, they cannot refine their neural controller to the real behavior of the robot.

We presents in next section experimental results of on-line learning because they are much more pertinent than results of off-line learning. Results establish that on-line learning scheme adapt the weights of the neural controller in order to solve the control objective.

3 Experimental validation of indirect backpropagation

3.1 Robot position control problem

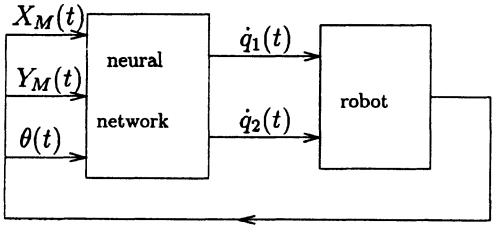
The problem is to control the cartesian position and the orientation of a two independent driving wheeled industrial mobile robot. Kinematic equations are done by equation (1) on figure 1. The main characteristic of this robot is its non-holonomy (see [7] and [4]).



$$\text{where} \begin{cases} \dot{X}_M &= \frac{r}{2}(\dot{q}_1 + \dot{q}_2)cos\theta \\ \dot{Y}_M &= \frac{r}{2}(\dot{q}_1 + \dot{q}_2)sin\theta \\ \dot{\theta} &= \frac{r}{2R}(\dot{q}_1 - \dot{q}_2) \\ r \text{ is} & \text{ radius of wheels} \\ R \text{ is} & \text{ half length of axis wheels} \end{cases} \quad (1)$$

Figure 1: Control problem and kinematic of the robot

The neural controller has to determine the instantaneous wheel velocities $\dot{q}_1$ and $\dot{q}_2$ that drive the robot at the desired configuration (X_d, Y_d, θ_d) where (X_d, Y_d) are the absolute desired cartesian coordinates of the middle point M of wheel axis ($X_d = Y_d = \theta_d = 0$). So, net inputs are cartesian and orientation errors and net outputs are the wheel velocities. The control structure is described on figure 2. Inputs of the network are normalized between -1 et $+1$ with respect to their maximum value ($D = 4$ meters is the half-longer of experimentation room).



$$\text{where} \begin{cases} X_M(t) \rightarrow X_{norm} &= \frac{X}{D} \\ Y_M(t) \rightarrow Y_{norm} &= \frac{Y}{D} \\ \theta(t) \rightarrow \theta_{norm} &= \frac{\theta}{\pi} \end{cases}$$

Figure 2: Neural control structure

3.2 Control criterion and learning gradient

The objective is to minimize the distance between the robot configuration and the desired one. Then the criterion is :

$$J = \alpha_1 X_M^2(t+1) + \alpha_2 Y_M^2(t+1) + \alpha_3(\theta(t+1) - \theta_s)^2 \quad (2)$$

where α_1, α_2, α_3 are normalization coefficients and $\theta_s = arctg(2\frac{Y_M(t)}{D})$ is a strategy constraint that helps the network on the singular configuration $X_M = 0$ and $Y_M \neq 0$.
By considering the time step Δt and the kinematic equations (1), the criterion becomes:

$$J = \alpha_1(X(t) + \Delta X(t+1))^2 + \alpha_2(Y(t) + \Delta Y(t+1))^2 + \alpha_3(\theta(t) + \Delta\theta(t+1) - \theta_s)^2 \quad (3)$$

$$\text{where} \begin{cases} \Delta X(t+1) &= \Delta U(t+1)cos(\theta(t) + \frac{\Delta\theta(t+1)}{2}) \\ \Delta Y(t+1) &= \Delta U(t+1)sin(\theta(t) + \frac{\Delta\theta(t+1)}{2}) \\ \Delta\theta(t+1) &= \frac{r}{2R}(\dot{q}_1(t) - \dot{q}_2(t))\Delta t \\ \Delta U(t+1) &= \frac{r}{2}(\dot{q}_1(t) + \dot{q}_2(t))\Delta t \end{cases}.$$

The gradient that trains the network is calculated with respect to the net outputs :

$$\frac{\partial J}{\partial \dot{q}_i(t)} = 2(\alpha_1 X(t+1)\frac{\partial \Delta X(t+1)}{\partial \dot{q}_i(t)} + \alpha_2 Y(t+1)\frac{\partial \Delta Y(t+1)}{\partial \dot{q}_i(t)} + \alpha_3(\theta(t+1) - \theta_s)\frac{\partial \Delta\theta(t+1)}{\partial \dot{q}_i(t)})$$

where:

$$\begin{cases} \frac{\partial \Delta X(t+1)}{\partial \dot{q}_i(t)} &= \Delta t\frac{r}{2}cos(\theta(t) + \frac{\Delta\theta(t+1)}{2}) - \frac{1}{2}\frac{\partial \Delta\theta(t+1)}{\partial \dot{q}_i(t)}\Delta U(t+1)sin(\theta(t) + \frac{\Delta\theta(t+1)}{2}) \\ \frac{\partial \Delta Y(t+1)}{\partial \dot{q}_i(t)} &= \Delta t\frac{r}{2}sin(\theta(t) + \frac{\Delta\theta(t+1)}{2}) + \frac{1}{2}\frac{\partial \Delta\theta(t+1)}{\partial \dot{q}_i(t)}\Delta U(t+1)cos(\theta(t) + \frac{\Delta\theta(t+1)}{2}) \\ \frac{\partial \Delta\theta(t+1)}{\partial \dot{q}_i(t)} &= \frac{r}{2R}\Delta t \ (i=1) \ or \ -\frac{r}{2R}\Delta t \ (i=2) \end{cases}$$

3.3 On line indirect backpropagation

We see in section 2 that our learning method permits to train the network off-line and on-line. The off-line learning method has been tested with good results. We solely presents in this section the results of the on-line learning method (Figure 3) because they are much more interesting. We argued that on-line learning scheme made feasible adaptive control.

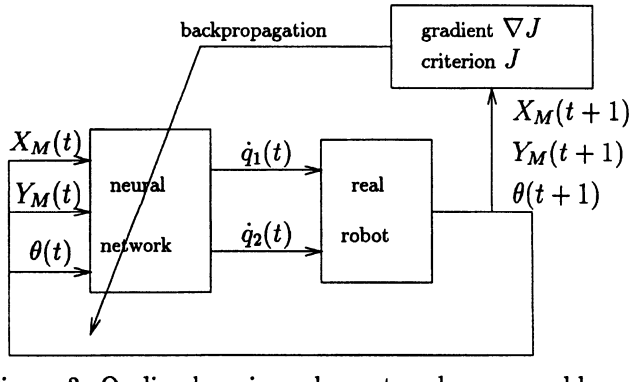


Figure 3: On-line learning scheme to solve our problem

As an illustration of that affirmation, we proposed to experiment on-line training with an **untrained network** which is updating after each control time step.

Protocol of experimentation

We have used a one hidden layer (of 9 neurons) network. The robot velocity is limited at 0.8 m/s during training. Only three training lessons have been necessary to train the network. At the beginning of each lesson, the robot is at an initial configuration far from the desired one (between 3 and 4 meters). The protocol is as follows:

- **Lesson 1:** The network is untrained (all the weights are randomly initialized). Initial robot configuration is $X_M = 3m$, $Y_M = 2m$, $\theta = 0°$.

- **Lesson 2:** Network has been trained by the first lesson. Initial robot configuration is still $X_M = 3m$, $Y_M = 2m$, $\theta = 0°$.
- **Lesson 3:** Network has been trained by the firsts lessons and Initial robot configuration is $X_M = 3m$, $Y_M = 0m$, $\theta = 180°$.

We stop each training when the robot completely stops.

First and second training

Right figure shows evolution of the training criterion. The learning duration is three times faster for the second training. Figure 4 shows cartesian trajectory, orientation of the robot, error and velocities of the wheels (i.e net outputs). For the first lesson, the robot trajectory is hazardous during ten minutes but we see clearly that the robot is attracted to the desired configuration.

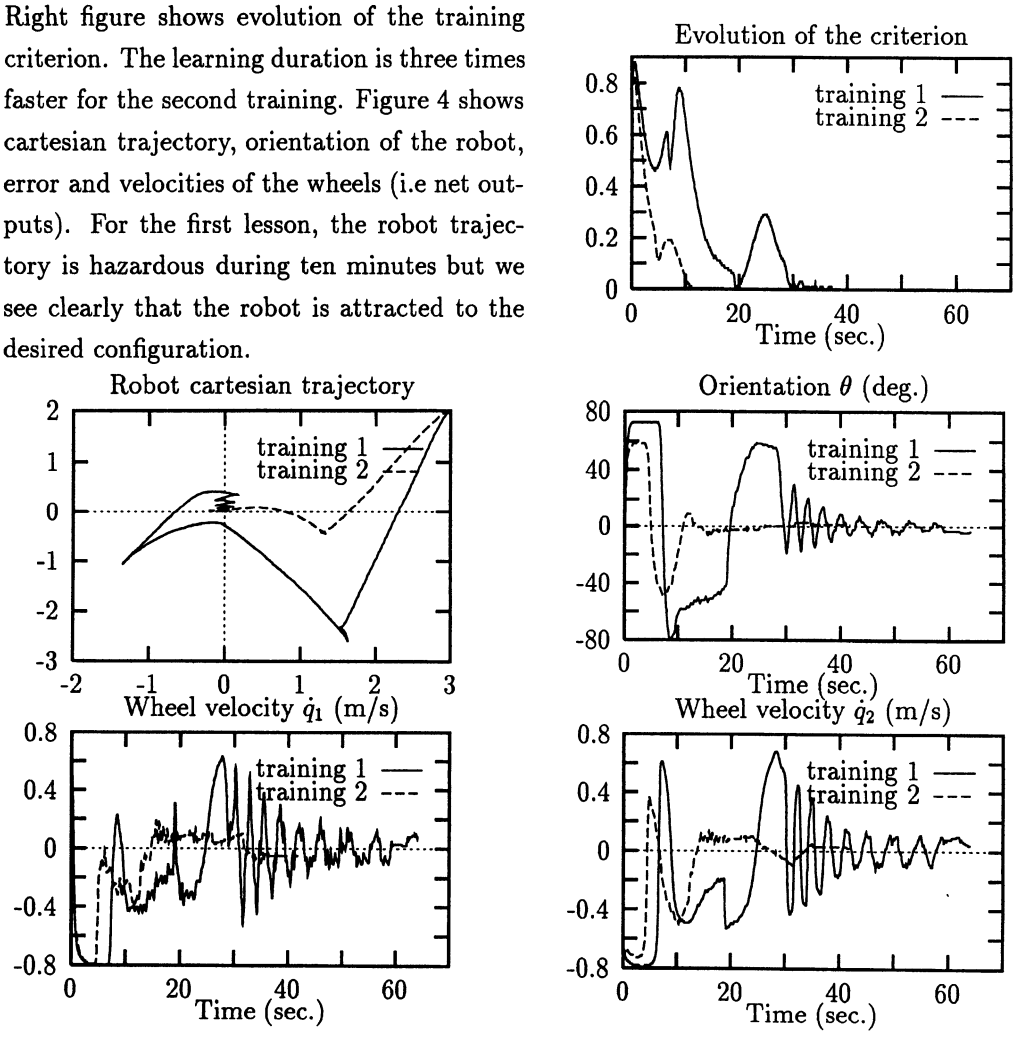


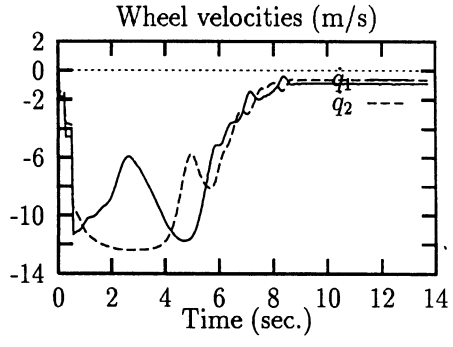
Figure 4: Trajectory (M point), orientation and control parameters of the robot

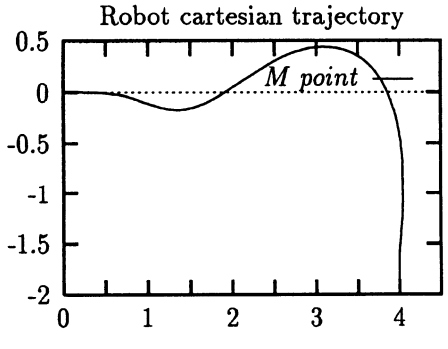
The trajectory is more direct and shorter in lesson 2 than in lesson 1. Final errors are reasonably good (about 1cm and 5 degrees) in regards with the robot size (1.025 m × 0.68 m) and are smaller after lesson 2. Wheel velocities are more smooth in lesson 2 and the average velocity greater.

The third lesson used a different initial configuration ($X_M = 3m$, $Y_M = 0m$, $\theta = 180°$). The results are similar to the previous and the robot reaches the desired configuration in 40 seconds. minutes (results are not plotted) .
These results shows that indirect backpropagation allows the neural-controller to identify the real behavior of the robot and determine a control law to drive the robot to the desired configuration and increase the quality of the control. Finally, we can argue that there is an knowledge acquisition of the robot kinematics and an adaption of the control.

Control of the real robot after on-line training

The trained network controls now the robot without learning from an initial configuration which has never been used for the learning: $X_M = 4m$, $Y_M = -2m$, $\theta = -90°$. The maximum velocity of the robot is 1.25 m/s. Right figure and figure 5 shows the net ability to control the robot with small final errors.





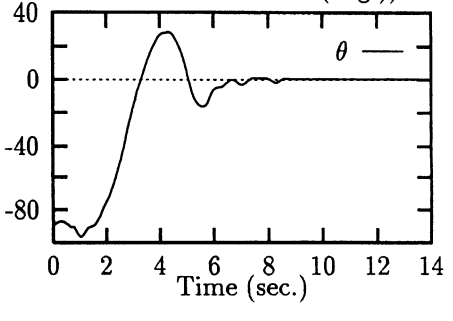


Figure 5: Control of the robot after three on-line trainings : trajectory and orientation

Others tests establish that the neural net controls the robot at any configuration in the experimentation room while $-3\ m \leq X_M \leq 3\ m$ and $-3\ m \leq X_M \leq 3\ m$.
These results show clearly that indirect backpropagation allows neural networks to learn control laws by especially specifying the control objective.

4 Conclusion

In this paper, we have presented experimentation results of an original approach to the Neural Network learning architecture for the control and the adaptive control of mobile robots. Our goal was to use multi-layer-networks and the backpropagation algorithm in

order to learn control objective without desired output. The principle of our approach is to express the control objective as a criterion. The gradient of the criterion is backpropagating through the network instead of the classical quadratic error. The technic permits the neuro controller to learn off-line or on-line. Experimental validation is realized by the position and the orientation control of a fast industrial mobile robot. To show the feasibility of the method, we trained a random neural network on the real robot with the on-line learning scheme. Results show clearly the very good and very fast adaption of the neural-network in front of the kinematics constraints and the dynamics effects of the robot.

References

[1] D.E Rummelhart, J.L. McClelland, *"Parallel Distributed Processing"*, MIT Press, pp. 318-362, 1986

[2] A. Guez, J. Selinsky, *A neuromorphic controller with a human teacher*, Proc. of IEEE International Conference on Neural Networks, pp 595-602, 1988

[3] D. Nguyen, B. Widrow *"Neural Networks for Self-Learning Control Systems"*,IEEE Work. on Industrial Applications of Neural Networks, pp 18-23,1991

[4] S. Delaplace, *"Navigation à Coût Minimal d'un Robot Mobile dans un Environnement Partiellement Connu"* thèse de l'Université Paris 6 (Pierre et Marie Curie), France, November 1991

[5] D.M.A. Lee, W.H. ElMaraghy, *"A Neural Network Solution for Biped Gait Synthesis"*, Int. Joint Conf. on Neural Networks, Vol II, pp. 763-767,1992

[6] M.I. Jordan and D.E Rummelhart, *"Forwars Models: Supervised Learning with a Distal Teacher"*, Cognitive Science, Vol 16, pp. 307-354, 1992

[7] C. K. Ait-Abderrahim, *"Commande de Robots Mobiles"*, Thèse de l'Ecole des Mines de Paris, 1993.

[8] P. Hénaff, H. Schwenk,M. Milgram *" A Neural Network Approach of the Control of Dynamic Biped Equilibrium"*, International Symposium on Measurement and Control in Robotics,1993

[9] P. Hénaff, *"Mises en Œuvre de Commandes Neuronales par Rétropropagation Indirecte: Application à la Robotique Mobile"*, thèse de l'Université Paris 6 (Pierre et Marie Curie), France, june 1994

[10] P. Hénaff, *"Adaptive Neural Control In mobile Robotics: Experimentation for a Wheeled Cart"*, IEEE Int. Conf. on Systems, Man and Cybernetics, pp 1139-1144, Oct. 1994