NEUROINTERFACES FOR HUMAN-MACHINE REAL TIME INTERACTION

Bernard Widrow[1], Edson P. Ferreira[2] and Marcelo M. Lamego[3]

Information Systems Lab., EE Dep., Stanford University

Abstract: This article aims at showing how neural networks can be employed in the generation of human-machine interfaces (neurointerfaces) for practical real time applications. In a great number of real world applications, due to technical and economic factors, full automation is not possible. In such cases, the human presence is essential and indeed, the system performance becomes highly dependent on human skills. Accordingly, an interface that modifies the problem, allowing unskilled human operators to perform the same task in a satisfactory way, becomes extremely useful. The adaptive nonlinear inverse modeling approach is employed as the basic methodology for specification and design of neurointerfaces. A successful application, a neurointerface that helps an operator to back up a scaled truck model connected to single-trailer and double-trailer configurations, is presented. *Copyright © 1998 IFAC*

Keywords: Neural Networks, Inverse Control, Adaptive Filters, Adaptive Control, Nonlinear Systems.

1. INTRODUCTION

Nowadays, the human-machine interaction has become a major concern. For many tasks, productivity, safety and liability conditions require a considerable degree of skills from human operators. In order to overcome the lack of skills, special human-machine neurointerfaces may be adopted. The basic idea is to change the operational space through a neural network, allowing the human operator to interact with the process through less-specialized commands. Accordingly, the operator devotes his attention to solve a less complex problem, directly at the task level. The objective is to improve the productivity, and safety levels of such tasks even in the case of unskilled operators.

This paper aims at showing how neural networks can be employed in the generation of human-machine interfaces for practical real time problems. Due to the complexity of such problems, neural networks are becoming a natural choice. Their abilities to reproduce highly nonlinear behaviors are described extensively in the literature.

In fact, the field of neural networks is experiencing a tremendous resurgence of activity. Successful industrial applications and favorable comparisons with conventional techniques are certainly the main causes of such interest. Fast growing low cost computational systems with considerable processing capabilities have provided an easy and reliable environment for the development of new training algorithms and more advanced topologies for neural network implementations.

This article is divided in 4 (four) sections. Section 2 presents the basic ideas concerning neurointerfaces. In Section 3, the adaptive inverse modeling approach, a framework utilized for neurointerface design, is described. A successful neurointerface application is presented in section 4. The neurointerface helps an operator to back up a scaled truck model connected to single-trailer and double-trailer configurations.

2. THE NEUROINTERFACE

A neurointerface may be thought of as an approximation of the system inverse model. Although such a statement may not be obvious, in fact, an operator develops with his experience a set of causal

[1] Professor.
[2] Visiting Professor sponsored by CAPES/UFES, Brazil.
[3] Ph.D. student sponsored by CNPq/UFES, Brazil.

rules that map standard behaviors into control actions (cognitive model), and this is exactly what inverse modeling does. Thus, a neurointerface tries to reproduce the actions of an experienced operator by using the system inverse model.

While cases might exist in which the neurointerface provides just an approximation of those actions taken by an expert operator, the change of operational space made by the neurointerface, although it may not solve the problem completely, allows the human operator to interact with the process through less-specialized actions. For instance, in a complex industrial process, such as a steel plant described in Vescovi, *et al.* (1997), the operators can not observe directly the main variables of interest during operation (quality, production level, etc.). Instead, they control the process by reasoning about a set of related variables that they can directly observe (pressure, speed, humidity, etc.). The relations between these observable variables and the main variables of interest are not known precisely. A neurointerface applied to such a process may not be able to completely invert the process model due to the lack of information, and consequently may not solve the interface problem completely. Nevertheless, the productivity, safety and liability conditions may be far improved with its utilization.

There are cases, however, where the neurointerface can be fully specified. The main variables of interest are either directly available or may be expressed as some function of the observed variables. In addition, the mapping between standard behavior and control actions can also be achieved by using knowledge of the functional relations between the main variables of interest. This is the case, for instance, in backing a truck and trailer to a loading platform. Although, this constitutes a difficult task for all but the most skilled truck drivers, a neurointerface can be fully specified and the trailer truck operation exercise reduced to a much less complex problem. In this case, the neurointerface may be considered as a black box that takes commands from the driver (desired direction of the trailer back part) and provides the necessary actions (steer the wheels) in order to achieve such a goal. The truck speed and the angle between cab and trailer are sufficient information to obtain a precise inverse modeling of the system. We should note that the driver was not eliminated in this problem. Nguyen and Widrow (1990) proposed a neural network that provided the full automation in backing a trailer truck to a loading dock and indeed, eliminating the presence of the driver. In the present work, the human action is essential. In fact, the driver is concerned with providing the desired spatial trajectory, free of obstacles and normally the shortest one.

The truck-backing-up exercise is a kinematics inverse modeling problem. It means that the dynamic effects that may occur during the operation are not significant. The neurointerface can also be applied to dynamic inverse modeling problems. A good example of this is the operation of a construction crane (Lamego and Rey, 1995). Since normally, a flexible cable does the coupling between the trolley car and the load, movements in the trolley car generate oscillations in the load. Thus, the crane operator is concerned in shifting the load from one point to another while achieving movement free of oscillations. Here, the neurointerface may be regarded as a black box that takes commands from the crane operator (desired trajectory of the load) and provides the necessary actions (actuation on each degree of freedom of the crane) in order to provide a smooth load movement.

It's also interesting to mention the case of robot arms because, in general, they do have inverse. For instance, Ferreira (1984) shows how to use the inverse nonlinear modeling to provide adaptive decoupling control for disturbance cancellation due to gravity, variable inertia and speed couplings (centrifuge and coriolis), in open chain robot arms. Using the Lagrange formalism the inverse model of an open chain robot arm is a very complex nonlinear vector equation. In general, there are many complaints about the complexity of this model. However, the inverse model has a nice hidden property: it is linear with respect to its parameters. Thus, the parameters can be easily identified and consequently, an inverse can fully specified. While this approach is general, a great analytical and computational effort is required to obtain a robot inverse model. In addition, each new robot requires a new model. A neurointerface applied to such a case may be able to completely invert the robot model, and consequently reduce the modeling effort. Here, the neurointerface may be considered as a black box that takes commands from the operator (desired spatial trajectory of the robot arm) and provides the necessary actions (actuation on each degree of freedom of the robot arm).

3. THE DESIGN OF NEUROINTERFACES USING ADAPTIVE NONLINEAR INVERSE MODELING

The nonlinear inverse modeling approach has been used for many years associated with linear feedback strategies. Basically, the objective is to cancel the plant nonlinear dynamical effects by using a nonlinear device that can reproduce an approximated inverse of the plant. The term "approximated" is employed to emphasize that, in general, a nonlinear system does not possess inverse. However, despite some pathological cases that might eventually exist, the methods of adaptive inverse modeling can often be applied to obtain acceptable inverse approximations of nonlinear systems.

The specification and design of a neurointerface use the fact that a nonlinear plant can be approximated by a neural network model, here represented by the function

$f : R^{(p+1)n+(q+1)m} \rightarrow R^n$, of the form

$$y_{k+1} = f(y_k, y_{k-1}, \cdots, y_{k-p}, u_k, u_{k-1}, \cdots, u_{k-q}, w_M)$$
$$y \in R^n, u \in R^m, w_M \in R^{\ell_M} \qquad (1)$$

Variables u and y are, respectively, the plant input and plant output vectors. w_M is the neural network weight vector.

The neurointerface can be regarded as a neural network approximation of the plant inverse model. It should be noted that the conventional and intuitive method, shown in figure 1, for adapting a linear filter to be the inverse of a linear plant does not apply to nonlinear cases. This is due to the fact that nonlinear systems do not commute. Roughly speaking, the block inversion shown in figure 1 does not work for nonlinear systems.

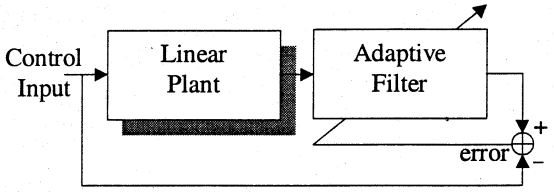


Figure 1. *Inverse modeling of a linear plant.*

Accordingly, the first step in a neurointerface design is to obtain a neural network model for the plant as defined in equation (1) and then, use it to obtain a neural approximation for the plant inverse (neurointerface). Figure 2 shows the nonlinear system identification procedure. The neural network uses the current and previous values of the plant input vector and also previous values of its output vector as its inputs. Its output represents an approximation of the plant output vector.

The neural model can be trained with a set of input-output data either acquired from the real plant or obtained from the plant mathematical model (if available).

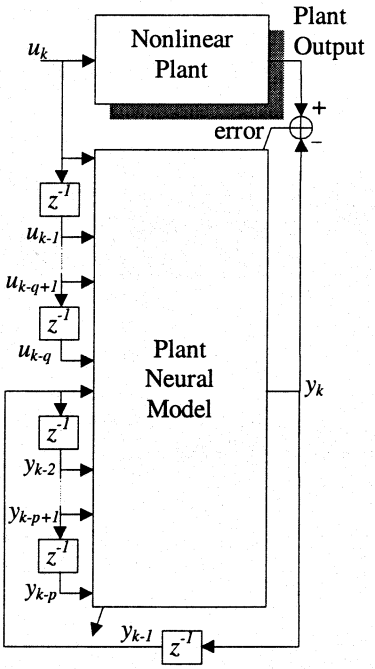


Figure 2. *Nonlinear system identification.*

The backpropagation-through-time, Werbos (1990), may be used to adapt the weights of the neural network model. If the input of the neural network model does not include any connection to plant output (a feedforward neural network), the conventional backpropagation algorithm, developed by Werbos (1974) and rediscovered and popularized by Rumelhart, et al. (1986), may be employed.

The final step, shown in figure 3, is to train the neurointerface to compute an approximated inverse for the obtained plant neural model.

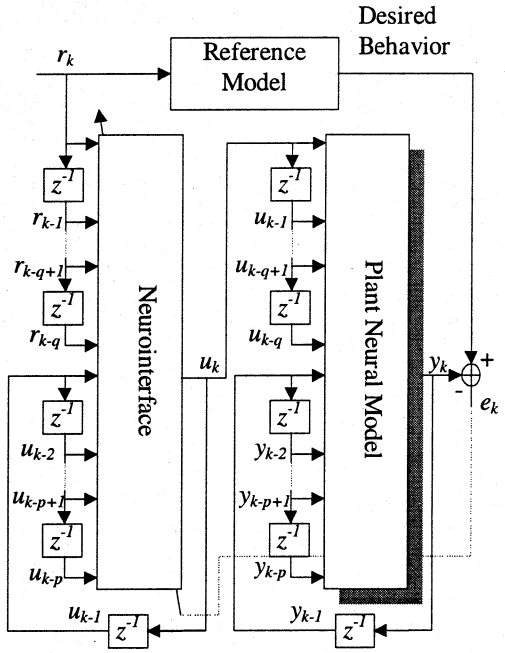


Figure 3. *Scheme used to adapt a neurointerface.*

Like the neural model case, the neurointerface computes a function $g : R^{(P+1)s+(Q+1)m} \rightarrow R^m$, of the form

$$u_k = g(u_{k-1}, u_{k-2}, \cdots, u_{k-Q}, r_k, r_{k-1}, \cdots, r_{k-P}, w_C)$$
$$u \in R^m, r \in R^s, w_C \in R^{\ell_C} \qquad (2)$$

Variables r and u are, respectively, the neurointerface input and output vectors and w_C is the neurointerface weight vector.

The vector r represents the ideal behavior of the plant neural model output vector y or a subset of it. Due to physical limitations, the ideal behavior, in most of cases, may not be achieved. Therefore, in order to provide a more realistic assumption concerning the desired behavior of the output vector y, a reference model can be adopted as shown in figure 3.

Because the plant is nonlinear, the neurointerface is trained in the configuration it will normally work with. The inversion shown in figure 1 is not allowed here. Consequently, to compute the mean square error

gradient with respect to the neurointerface weights, information concerning the plant must be available. This is the reason why the plant is replaced by its neural realization during the neurointerface adaptation procedure.

The change in the neurointerface weights at each training step is in the negative direction to the gradient of the system error (e_k). To find the gradient, the chain-rule expansion for ordered derivatives is employed

$$\frac{\partial \|e_k\|^2}{\partial w_C} = -2 \frac{\partial^+ y_k}{\partial w_C} e_k \tag{3}$$

with

$$\frac{\partial^+ y_k}{\partial w_C} = \sum_{i=0}^{q} \left(\frac{\partial y_k}{\partial u_{k-i}} \right) \left(\frac{\partial^+ u_{k-i}}{\partial w_C} \right) + \sum_{i=1}^{p} \left(\frac{\partial y_k}{\partial y_{k-i}} \right) \left(\frac{\partial^+ y_{k-i}}{\partial w_C} \right) \tag{4}$$

and

$$\frac{\partial^+ u_k}{\partial w_C} = \left(\frac{\partial u_k}{\partial w_C} \right) + \sum_{i=1}^{Q} \left(\frac{\partial u_k}{\partial u_{k-i}} \right) \left(\frac{\partial^+ u_{k-i}}{\partial w_C} \right) \tag{5}$$

Each of the terms in equations (4) and (5) is either a Jacobian matrix, which may be calculated using the *dual-subroutine* (Werbos, 1992) of the backpropagation algorithm, or is a previously calculated value of $\partial^+ u_k / \partial w_C$ or $\partial^+ y_k / \partial w_C$. To be more specific, the first term in equation (5) is the partial derivative of the neurointerface's output with respect to its weights. This term is one of the Jacobian matrices of the neurointerface and may be calculated with the dual subroutine of the backpropagation algorithm. The second part of equation (5) is a summation. The first term of the summation is the partial derivative of the neurointerface's current output with respect to a previous output. However, since the neurointerface is externally recurrent, this previous output is also a current input. Therefore the first term of the summation is really just a partial derivative of the output of the neurointerface with respect to one of its inputs. By definition, this is a sub-matrix of the Jacobian matrix for the network, and may be computed using the dual-subroutine of the backpropagation algorithm. The second term of the summation in equation (5) is the ordered partial derivative of a previous output with respect to the weights of the neurointerface. This term has already been computed in a previous evaluation of equation (5), and need not be re-computed. A similar analysis may be performed to determine all of the terms required to evaluate equation (4). After calculating these terms, the weights of the neurointerface may be adapted using the weight-update equation

$$\Delta w_{C\,k} = 2\mu \frac{\partial^+ y_k}{\partial w_C} e_k \tag{6}$$

The neurointerface is designed to operate in real time without any adaptation algorithm. This implies that its adaptation procedure is performed offline. Figure 4 shows the basic configuration that a neurointerface is supposed to work with. Note that there is no external feedback in this topology. The neurointerface does not cancel disturbances that may occur in the plant. It just changes the operational space through a recurrent neural network, allowing the human operator to interact with the process through less-specialized commands.

Nonetheless, there are situations where the plant inversion supplied by the neurointerface is not enough to provide reliable operating conditions. Disturbance effects may occur during the plant operation and may lead it to risky operating regions. To overcome the disturbance effects and provide more reliable operating conditions to the operator, adaptive linear control schemes may be adopted.

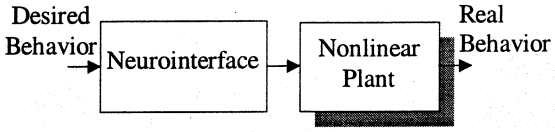


Figure 4. *Basic topology.*

Figure 5 provides a conventional adaptive linear control topology that can be used with a neurointerface. In this case, the adaptive linear controller may have its parameters adapted using the same reference model adopted to the neurointerface training as a plant idealization. Of course, the reference model has to be linear and differentiable. The last restriction allows the error gradient computation with respect to the adaptive linear controller parameters and indeed, the adaptation of the linear controller.

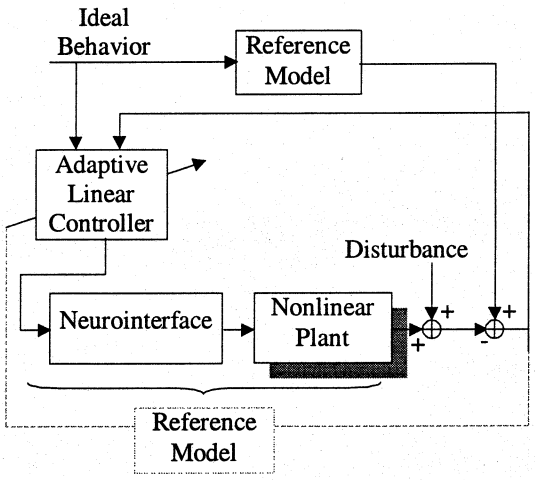


Figure 5. *Neurointerface working in closed loop.*

Another possible adaptive linear control scheme is shown in figure 6. It is the adaptive inverse control technique described in Widrow and Wallach (1996). In this scheme, an equivalent linear model for the neurointerface and the nonlinear plant combined is identified in real time. Then, using a digital copy of the linear model, the linear controller C and the linear

disturbance canceler Q are calculated offline. The offline process can rum much faster than the real time, so that as the plant linear model is evaluated, Q and C are immediately obtained.

The same procedure used in adaptive linear systems can be employed here. The neurointerface is able to cancel most of the nonlinear effects the plant may have and indeed, it can be used in combination with adaptive linear control schemes for the control of nonlinear plants.

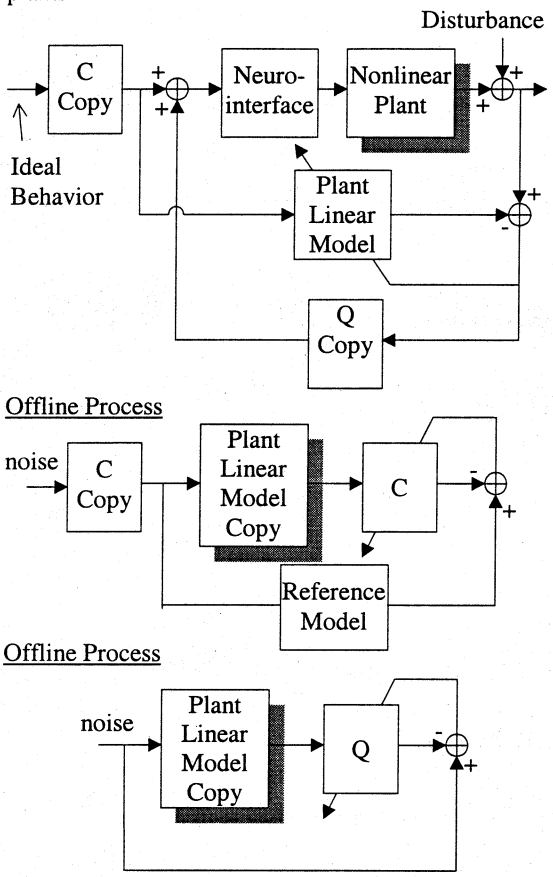


Figure 6. *Neurointerface combined with adaptive inverse linear control.*

4. EXPERIMENTAL RESULTS

Backing a truck and trailer to a loading platform constitutes a difficult task for all but the most skilled truck drivers. This section briefly presents the experimental results of a neurointerface that reduces the trailer truck operation exercise to a much less complex problem. Two configurations are considered: a scaled truck model connected to single-trailer configuration (see figure 7) and connected to a double-trailer configuration (see figure 8). The neurointerface may be considered as a black box that takes commands from the driver (desired direction of the trailer back part) and provides the necessary actions (steer the front wheels). For the single-trailer configuration, the desired direction of the trailer back part is related to the angle between cab and trailer. For the double-trailer configuration, it is related to the angle between the first trailer and the second one. In both implementations, the neurointerface works in closed loop. The adaptive linear control topology presented in section 3, figure 6 is employed here. For the single trailer configuration, the neurointerface has, as its inputs, the truck speed, the desired value of the angle between cab and trailer (θ_2) and the previous value of the neurointerface's output (the front wheel steering angle, θ_1). Similarly, for the double-trailer configuration, the neurointerface has as its inputs, the truck speed, the angle θ_2, the desired value of the angle between the first trailer and the second one (θ_3) and the previous value of the neurointerface's output (the front wheel steering angle, θ_1). The neurointerface was designed following the steps described in section 3. Acquired data from the truck prototype were used to obtain the neural model for each configuration. The obtained neural models were used for the training of the neurointerfaces. In both cases, only the disturbance canceler Q, an adaptive linear combiner, is utilized. The involved dynamics are simple and, indeed, the controller C need not be implemented. The equivalent plant linear model is adapted in real time.

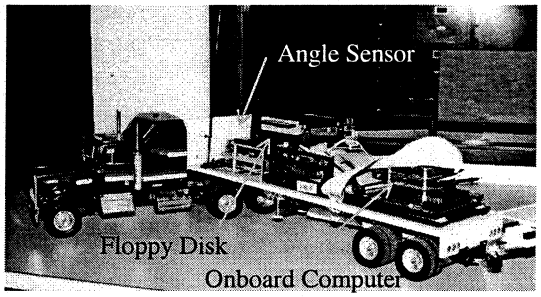


Figure 7. *Truck model connected to a single-trailer configuration.*

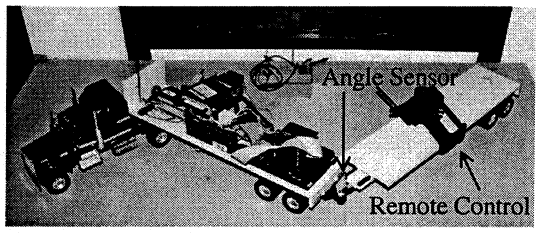


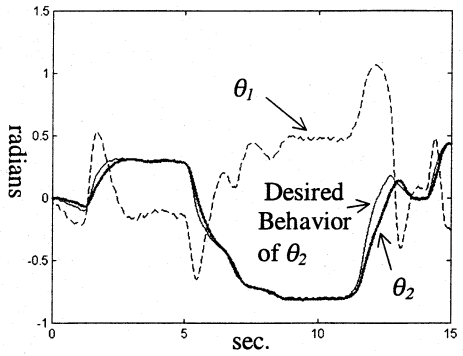
Figure 8. *Truck model connected to double-trailer configuration.*

The offline process for adaptation of Q is started at every 500 samples, sampling period of 30 ms. The data acquired in this interval (15 sec.) are used as the training set for Q. The experimental results are shown in figures 9 and 10. They correspond to sequences of data acquired from the truck model moving backwards in both configurations.

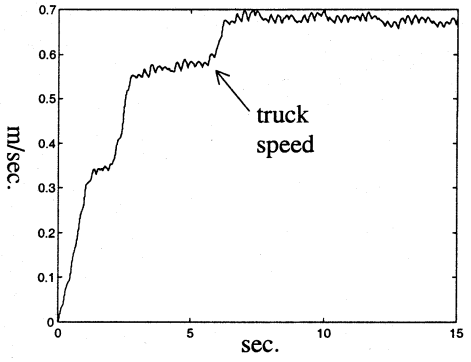
CONCLUSIONS

This article presents a new approach for generation of human-machine interfaces through neural networks (neurointerface) for practical real time applications. The adaptive nonlinear inverse modeling approach is

employed as the basic methodology for specification and design of neurointerfaces. The neurointerface is able to cancel most of the nonlinear effects the plant may have and, indeed, it can be used in combination with adaptive linear control schemes for the control of nonlinear plants. A successful application, a neurointerface that helps an operator to back up a scaled truck model connected to single-trailer and double-trailer configurations, is also presented. This is an introductory work and a great effort will be needed to improve the neurointerface approach. However, the general aspects covered in this paper combined with the excellent quality of the experimental results lead to conclude that the fully utilization of neurointerfaces for real time applications seems to be very promising.

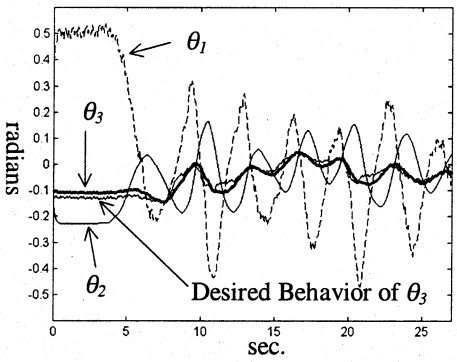


(a)

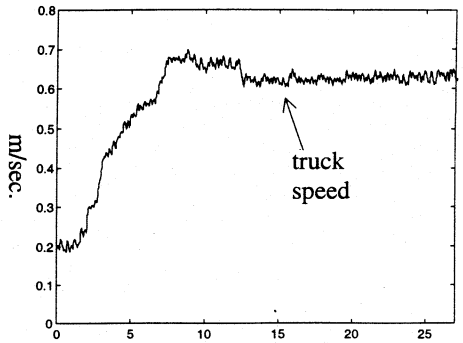


(b)

Figure 9. *Experimental results for the truck model connected to a single-trailer configuration: (a) desired behavior of θ_2, real behavior of θ_2 and neurointerface's output (θ_1); (b) truck speed.*



(a)



(b)

Figure 10. *Experimental results for the truck model connected to a double-trailer configuration: (a) desired behavior of θ_3, real behavior of θ_3 and θ_2 and neurointerface's output (θ_1); (b) truck speed.*

REFERENCES

Ferreira, E. P. (1984). *Contribution a L'Identification de Parametres et a la Commande Dynamique-Adaptative des Robots Manipulateurs.* Dr. thesis, LAAS du CNRS, Universite Paul Sabatier, Toulose, France, July.

Lamego, M. M. and J. P. Rey (1995). *The Interval Based Control Technique: Controlling Physical Systems Through Imprecise Models.* Proceedings of the IEEE Industrial Applications Society Annual Meeting, Florida, USA, October, pp. 1822–1827.

Nguyen, D. and B. Widrow (1990). *Neural Networks for Self-learning Control Systems.* IEEE Control Systems Magazine, April, pp. 18–23.

Rumelhart, D. E., G. E. Hinton and R. J. Williams (1986). *Learning internal representations by error propagation.* Parallel Distributed Processing. (D. E. Rumelhart and J. L McClelland editors), volume 1, chapter 8. MIT Press, Cambridge, MA.

Vescovi M. R., M. M. Lamego and A. Farquhar (1997). *Modeling and Simulation of a Complex Industrial Process.* IEEE Expert Magazine — Intelligent Systems & Their Applications, IEEE Computer Society, May/June, pp. 42-46.

Werbos, P. (1974). *Beyond Regression: New Tools for Prediction and Analysis in the Behavioral Sciences.* PhD thesis, Harvard University, Cambridge, MA, August.

Werbos, P. (1990). *Backpropagation through time: What it does and how to do it.* Proceedings of the IEEE, Vol. 78(10), October, pp. 1545–1680.

Werbos, P. (1992). *Neurocontrol and supervised learning: An overview and evaluation.* Handbook of Intelligent Control: Neural, Fuzzy and Adaptive Approaches. (D. White and D. Sofge editors), chapter 3. Van Nostrand Reinhold, New York.

Widrow, B. and E. Walach (1996). *Adaptive Inverse Control.* Prentice Hall PTR, Upper Saddle River, NJ.