View metadata, citation and similar papers at core.ac.uk

brought to you by CORE

provided by IntechOpen

We are IntechOpen,
the world's leading publisher of
Open Access books
Built by scientists, for scientists

4,800
Open access books available

122,000
International authors and editors

135M
Downloads

154
Countries delivered to

Our authors are among the
TOP 1%
most cited scientists

12.2%
Contributors from top 500 universities

CLARIVATE ANALYTICS
BOOK CITATION INDEX
INDEXED

WEB OF SCIENCE™

Selection of our books indexed in the Book Citation Index
in Web of Science™ Core Collection (BKCI)

Interested in publishing with us?
Contact book.department@intechopen.com

Numbers displayed above are based on latest data collected.
For more information visit www.intechopen.com

Multi-Robot Systems Control Implementation

José Manuel López-Guede, Ekaitz Zulueta,
Borja Fernández and Manuel Graña
Computational Intelligence Group, University of the Basque Country (UPV/EHU)
Spain

1. Introduction

Nowadays it is clear that multi-robot systems offer several advantages that are very difficult to reach with single systems. However, to leave the simulators and the academic environment it is a mandatory condition that they must fill: these systems must be economically attractive to increment their implantation in realistic scenarios. Due to multi-robots systems are composed of several robots that generally are similar, if an economic optimisation is done in one of them, such optimisation can be replicated in each member of the team.

In this paper we show a work to implement low level controllers with small computational needs that can be used in each of the subsystems that must be controlled in each of the robots that belongs to a multi-robot system. If a robot is in a multi-robot system that robot needs bigger computational capacity, because it has to do some tasks derived from being in the team, for example, coordination and communication with the remaining members of the team. Besides, occasionally, it has to deduce cooperatively the global strategy of the team. One of the theoretical advantage of multi-robot systems is that the cost of the team must be lower than the cost of a single robot with the same capabilities. To become this idea true it is mandatory that the cost of each member was under a certain value, and we can get this if each of them is equipped with very cheap computational systems. One of the cheapest and more flexible devices for control systems implementation are Field Programmable Gate Arrays (FPGAs). If we could implement a control loop using a very simple FPGA structure, the economic cost of each of them could be about 10 dollars.

On the other hand, and under a pessimistic vision, the subsystems to control could have problems to be controlled using classic and well known control schemas as PID controllers. In this situation we can use other advanced control systems which try to emulate the human brain, as Predictive Control. This kind of control works using a world model and calculating some predictions about the response that it will show under some stimulus, and it obtains the better way of control the subsystem knowing which is the desired behavior from this moment until a certain instant later. The predictive controller tuning is a process that is done using analytical and manual methods. Such tuning process is expensive in computational terms, but it is done one time and in this paper we don’t deal with this problem.

However, in spite of the great advantage of predictive control, which contributes to control systems that the classic control is unable to do, it has a great drawback: it is very computationally expensive while it is working. In section 4 we will revise the cause of this

problem. A way of avoiding this drawback is to model the predictive controller using neural networks, because once these devices are trained they perform the calculus at great speed and with very small computational requirements, and at the same time, we can implement them using very cheap FPGA devices. In this paper we propose a learning model to be used with Time Delayed Neural Networks, so once the neural network is trained, the neuronal predictive controller is ready and it responds properly showing its generalization capabilities in environments that it hasn't seen in the training phase. This way we could get a very cheap implementation of each of the control loops that each robot of the multi-robot team needs, avoiding the rise of the total cost of the team.
In the literature there are several sources indicating that each robot of a multi-robot system must be as cheap as possible. There is a quantitative support for the argument that larger teams of less-reliable and cheaper robots can perform certain missions more reliably than smaller teams of more-reliable robots (Stancliff et al., 2006). There are examples of using very cheap discrete components. In (O'Hara & Balch, 2007) very cheap sensorless nodes are used to support a complex multi-robot foraging task. On the other hand, in (Wu et al., 2008) a kind of sensors is used because they became cheaper that others. In (Kornienko et al., 2005), the components of the developed system consume energy provided by microcontroller's I/O ports, are cheap and available on micro-component market. Besides the use of individual components, (Andrews et al., 2007) integrate economic and technical issues into an unified engineering design framework for the manufacturers of robots. There are examples that have been done as previous works in the same direction that this paper (López-Guede et al., 2008).
Section 2 gives a summary of the objective of the work of this paper. Section 3 gives background information about Predictive Control and a technique called Dynamic Matrix Control, and about a kind of neural nets called Time Delayed Neural Networks. Section 4 discusses a concrete case study and the results that we obtain. Finally, conclusions are covered in section 5.

2. Objective

The main objective of this paper is to get cheap implementation of low level control loops that could be used by each member of a multi-robot system. To get this objective Time Delayed Neural Networks are used to model predictive controllers, because these can control subsystems that classics controllers can't.

3. Background

This section gives a brief introduction about a general technique called Model Predictive Control, and about a concrete technique called Dynamic Matrix Control. We also present a brief introduction about Time Delayed Neural Networks.
We consider that it is necessary to understand the advantages of this kind of control, that make it very useful in some circumstances, and their drawbacks, and then understand how a neural network based implementation can eliminate these drawbacks.

3.1 Model Predictive Control (MPC)

Model Predictive Control (MPC) is an advanced control technique used to deal with systems that are not controllable using classic control schemas as PID. This kind of controllers works like the human brain in the sense that instead of using the past error between the output of the system and the desired value, it controls the system predicting the value of the output in a short time, so the system output is as closer as possible to its desired value for these moments. Predictive Control isn’t a concrete technique. It’s a set of techniques that have several common characteristics: there is a world model that is used to predict the system output from the actual moment until *p* samples, an objective function that must be minimized and a control law that minimizes the objective function. The predictive controllers follow these steeps:

- Each sampling time, through the system model, the controller calculates the system output from now until *p* sampling times (prediction horizon), which depends on the future control signals that the controller will generate.
- A set of *m* control signals is calculated optimizing the objective function to be used along *m* sampling times (control horizon).
- In each sampling time only the first of the set of *m* control signals is used, and in the next sampling time, all the process is repeated again.

The concept of Predictive Control is a set of techniques that share certain characteristics, and the engineer has liberty to choose in each of them. So, there are several types of predictive controllers. These characteristics are the following:

- There is a plant model, and there can be used a step response model, an impulse step response model, a transfer funcion, etc.
- There is an objective funcion that the controller has to optimize.
- There is a control law to minimize the objective function.

To learn more about Predictive Control in general and about diverse predictive control algorithms, see (Camacho & Bordons, 2004), (Camacho & Bordons, 1995), (Maciejowski, 2002), (Sunan et al., 2002), (Rawlings, 1999) and (Soeterboek, 1992).

3.1.1 Model predictive control advantages

From a theoretical point of view, model predictive based controllers have some advantages:

- It is an open methodology, with possibility of new algorithms.
- They can include constraints on manipulated variables as well as on controlled variables. This is important to save energy and to get the working point as near as possible from the optimum.
- They can deal with multivariable systems in a simplest way than other algorithms.

From a practical point of view, model predictive based controllers have the advantage that they can deal with systems that show stability problems with classical control schemes. To show this property we will use one of these systems in Section 4.

3.1.2 Model predictive control drawbacks

The main drawback of predictive controllers isn’t that it was very expensive in computationally terms in the tuning phase, because it is carried out only one time. The main drawback is that the computational requirements of the shown controller are great when it’s in its working phase. Each sample time the controller must calculate the control law of the equation (1), and there are involved several matrix operations, as several multiplication, an addition and a subtraction. Performing these operations we obtain a set of *m* control signals, but only the first of them is used in this sample time, the rest are ignored. The algorithm woks in this way, but it is computationally inefficient.

3.2 Dynamic Matrix Control (DMC)

The technique called Dynamic Matrix Control (DMC) is a concrete MPC algorithm that fixes each of the three characteristics that we have seen inf the following way.
To learn more about Dynamic Matrix Control in particular, see (Camacho & Bordons, 2004), (Camacho & Bordons, 1995), (Maciejowski, 2002) and (Sunan et al., 2002).

3.2.1 Subsystem model

The plant model used by DMC algorithm is the step response model. This model obtains the g_i coefficients that are the output of the lineal subsystem when it is excited using a step. To reduce the number of coefficients we assume that the subsystem is stable and the output doesn't change after a certain sampling time k.

$$y(t) = \sum_{i=1}^{k} g_i \Delta u(t-i) \tag{1}$$

3.2.2 Prediction model

Using the step response model to model the subsystem and maintaining the hypothesis that perturbations over the subsystem are constants, it is possible to calculate a prediction in the instant t of the output until the instant ($t + p$) under the effect of m control actions:

$$\hat{y} = G\Delta u + f \tag{2}$$

being $\hat{y}$ the prediction of the output, G a matrix that contains the subsystem's dynamics and f the free response of the subsystem. Following we show the dimension of this matrix and these vectors:

$$\hat{y} = \begin{bmatrix} \hat{y}(t+1\,|\,t) \\ \hat{y}(t+2\,|\,t) \\ \vdots \\ \hat{y}(t+p\,|\,t) \end{bmatrix}_p \quad G = \begin{bmatrix} g_1 & 0 & \cdots & 0 \\ g_2 & g_1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ g_m & g_{m-1} & \cdots & g_1 \\ \vdots & \vdots & \ddots & \vdots \\ g_p & g_{p-1} & \cdots & g_{p-m+1} \end{bmatrix}_{pxm}$$
$$\Delta u = \begin{bmatrix} \Delta u(t) \\ \Delta u(t+1) \\ \vdots \\ \Delta u(t+m-1) \end{bmatrix}_m \quad f = \begin{bmatrix} f(t,1) \\ f(t,2) \\ \vdots \\ f(t,p) \end{bmatrix}_p \tag{3}$$

In the equation (5) we show how the free response of the subsystem $f(t,k)$ is calculated:

$$f(t,k) = y_m(t) + \sum_{i=1}^{N} (g_{k+i} - g_i)\Delta u(t-i) \tag{4}$$

3.2.3 Control law

The obtention of the control law is based on the existence of an objective function, which uses the future outputs prediction model that we have described before. As objective function we used the described by this equation:

$$J = \sum_{j=1}^{p} \left[\hat{y}\left(t+j \mid t\right) - w\left(t+j\right) \right]^2 + \sum_{j=1}^{m} \lambda \left[\Delta u\left(t+j-1\right) \right]^2 \tag{5}$$

We have to minimize the difference between the reference and the output prediction along a prediction horizon p with the m control actions generated in the control horizon, ponderating the rougness in the variation of the manipulated variables using λ. Minimizing the objective funcition J described in the equation (5) we obtain the folowing expression, that produces m control actions, although in t only one of the is used:

$$\Delta u = \left[\left(G^t G + \lambda I \right)^{-1} G^t \left(w - f \right) \right]_m \tag{6}$$

3.3 Neuronal implementation

Following with the discussion about the computationally inefficiency of the analytical predictive control shown in the previous section, we think that it would be convenient to have a mechanism that could implement such controller requiring less computational power. An alternative to get this is to use neural networks, and more precisely, Time Delayed Neural Networks, because as the rest of neural networks, they are very fast, computationally inexpensive and they have the ability of generalizing their responses.
This section gives a brief introduction about a kind of neural networks called Time Delayed Neural Networks, that we have used to model the previous model predictive controller to eliminate the shown drawbacks. To learn more about neural networks in general see (Braspenning et al., 1995), (Chester, 1993) and (Widrow & Lehr, 1990). To learn more about identification and control of dynamical systems, see (Narendra & Parthasarathy, 1990) and (Norgaard et al., 2003), and about neural identification applied to predictive control see (Arahal et al., 1998) and (Huang, 2000). There are interesting approximations to the prediction capacity of neuronal models when predictive control is present, (McKinstry et al., 2006), (Aleksic et al., 2008) and (Kang, 1991). Stability of these neural networks is an important issue, (Wilson, 1995).

3.3.1 Time Delayed Neural Networks

Time Delayed Neural Networks (TDNN) are a kind of multi-layer perceptron neural networks. They have an input layer, where are the neurons that accept the exterior inputs; one or more hidden layers, that generate intermediate values; and a final layer, that puts outside the data generated by the net. The TDNN special feature is that they are a kind of dynamic neural networks, because delayed versions of the input signals are introduced to the input layer. Due to this, the outputs don't depend only on the actual values of the signals, they depend on the past value of the signals too. This is one of the main parameters of a TDNN: the size of the delay line. The TDNNs that are going to be used in this work only have forward conections, so they aren't neither recurrent nor partially recurrent. This kind of neural network can be trained using the Backpropagation algorithm or the Generalized Delta Rule. In the experiments that we show in this paper the Levenberg-

Marquardt method has been used. To learn more about Time Delayed Neural Networks, see (Huang et al., 2000), (Huang et al., 2006), (Waibel et al., 1989), (Wang et al., 2005) and (Taskaya-Temizel & Casey, 2005).

3.3.2 Structural parameters

As we are worried about the computational cost of our implementation of the predictive controller, we have limited the number of hidden layers to one, so we assume that we are working with a time delayed neural network that has the simplest structure. Once we have established this constraint, the main parameters that configure the structure of this TDNN are the number of neurons of the hidden layer and the size of the time delay line, in other words, the number of delayed versions of the input signals are introduced to the input layer. We will try to get these parameters as small as possible to minimize the computational cost of the resultant implementation. The last main parameter to establish is the kind of the function that will be executed in each neuron, and we will take into account that the linear function is the least expensive from the computational point of view. In Fig. 1 we show the structure of the Time Delayed Neural Network that we have used to get our purpose, in which we have fitted the size of time delay line *d* and the size of hidden layer *h* parameters.

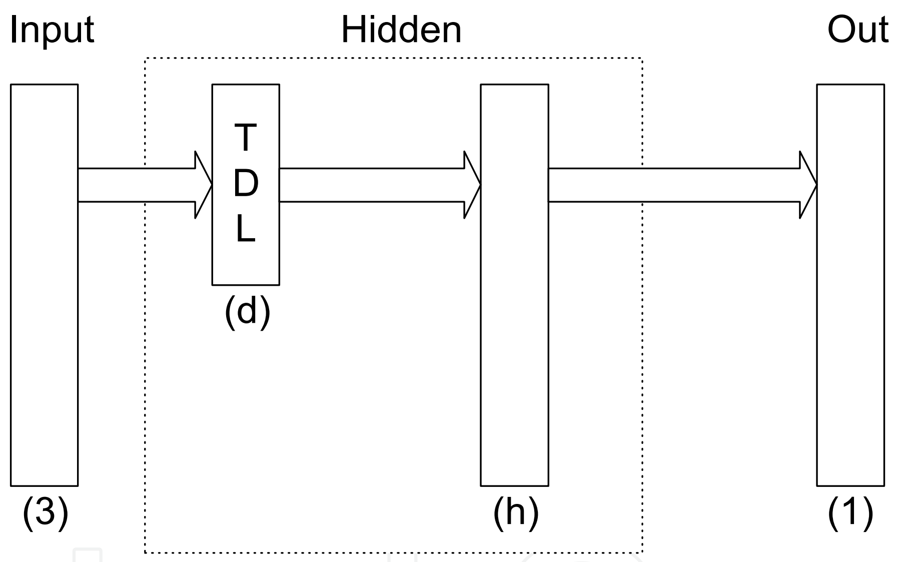


Fig. 1. Time Delayed Neural Network structure with 3 layers: input layer with 3 inputs, and the output layer with 1 output.

4. Case study

In this case study we will show an example of how we can get an advanced control of robot subsystems using neural networks. We are going to suppose that the model of a subsystem is described by the following discrete transfer function:

$$H(z) = \frac{1}{z - 0.5} \tag{7}$$

As we can see analyzing the poles, this subsystem is stable. Although it is a stable subsystem, its response is unstable if we try to control it using a discrete PID controller tuned through classic and well-known method as Ziegler-Nichols, as we can see in Fig. 2.

Once we have seen that the respone is unstable, we decide to use Model Predictive Control. To work with this kind of control we have to stablish the working point. To do this, we examine the Bode diagram of the Fig. 3 and we choose the frecuency of the marked point of this figure.
Once we have determined the working point in Fig. 3, we design the reference signal. As it is shown in Fig. 4, using a properly tuned DMC predictive controller, for example, with the values for its parameters $p = 5$, $m = 3$ y $\lambda = 1$, a right control is obtained.
To get this control it has been mandatory to tune the DMC controller. This phase is very expensive in computationally terms, but it's carried out only one time. However, the computational requirements of DMC controller are great when it's in its working phase, due to the operations that it must perform to get the control law, and although it obtains set of m control signals, only first of them is used in this sample time, the rest are ignored. Because of this, it would be convenient to have a mechanism that could implement such controller requiring less computational power. Besides, it may be necessary to control several subsystems of this kind in each robot of the multi-robot team. An alternative to get this is to use neural networks, and more precisely, Time Delayed Neural Networks, because, as the rest of neural networks, they are very fast and they have the ability of generalizing their responses.
In the literature there are works comparing PID and MPC controllers (Voicu et al., 1995).
Now we deal with the concrete problem of getting a neuronal predictive controller that could control the system described by the discrete transfer function of the equation (7) using Time Delayed Neural Networks.

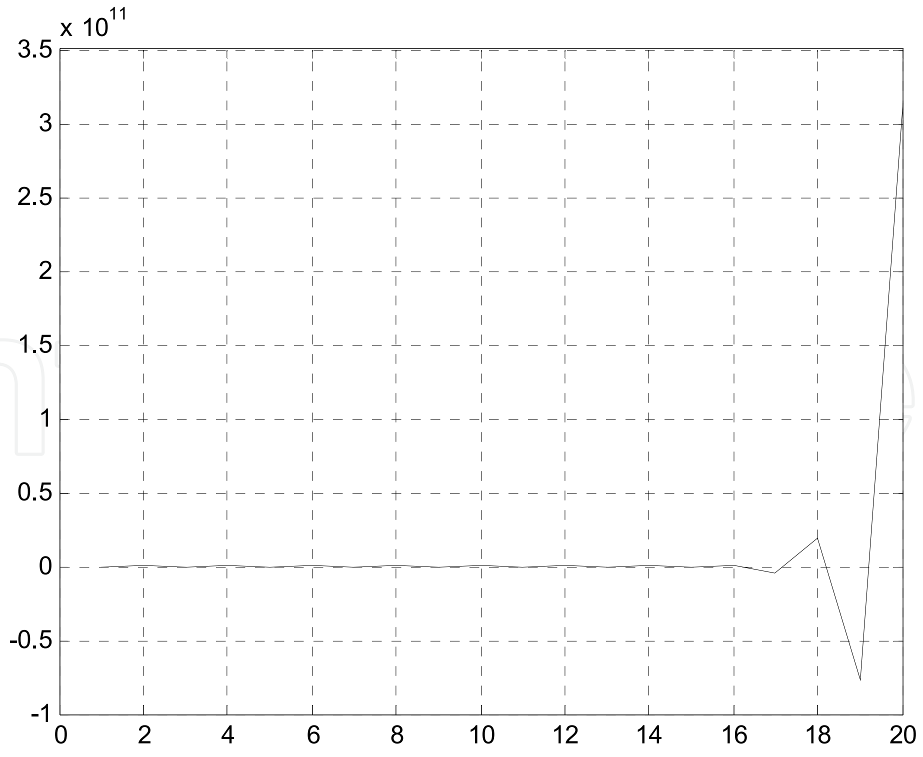


Fig. 2. Unstable response of the subsystem under the control of a discrete PID controller.

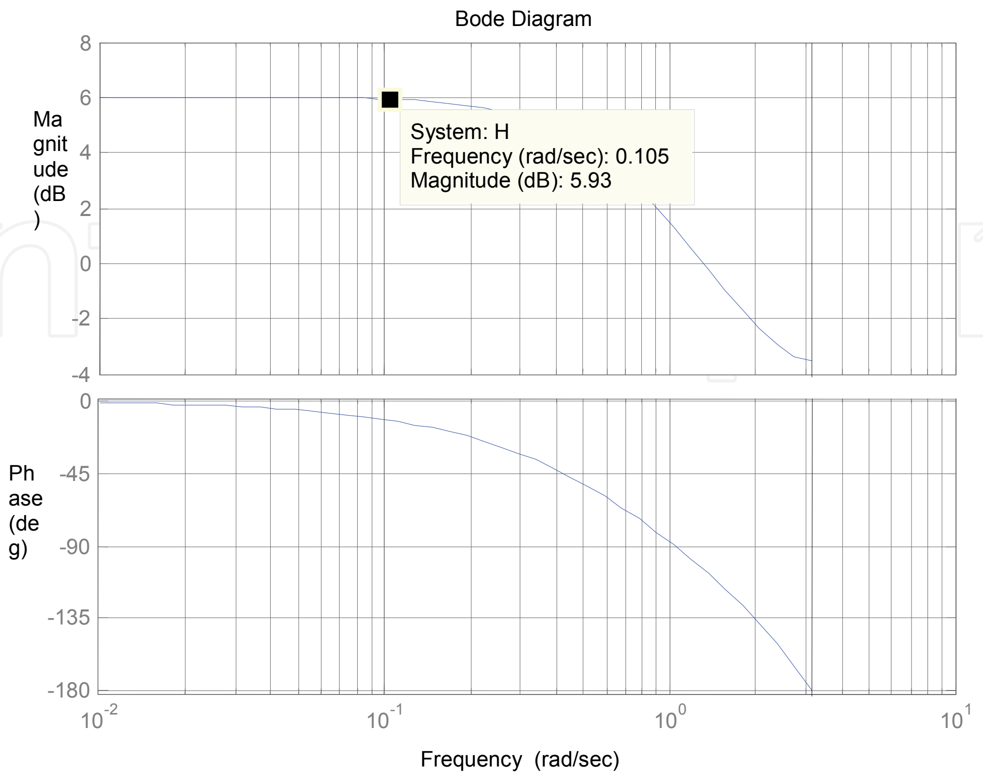


Fig. 3. Bode diagram of the subsystem, showing the chosen point.

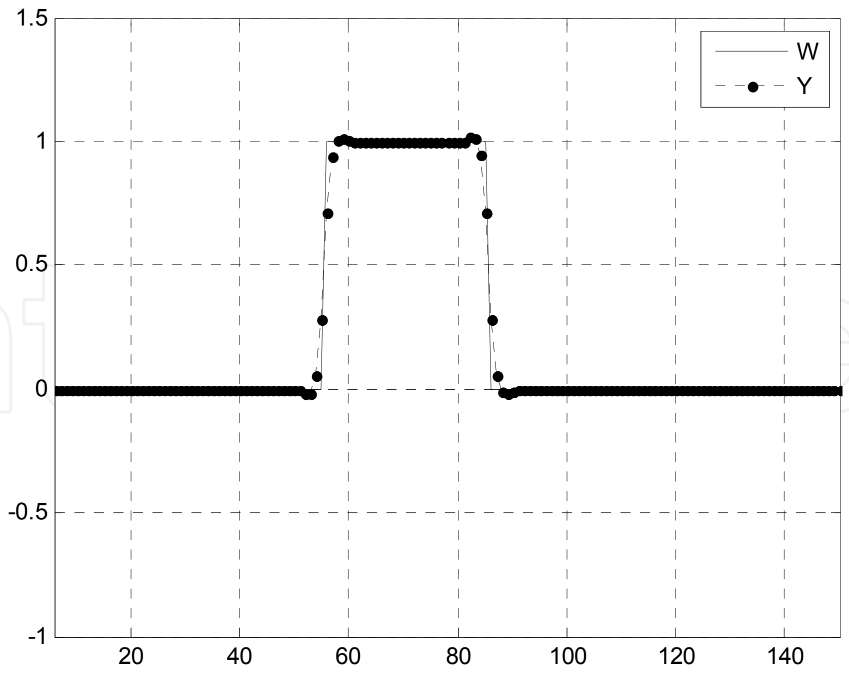


Fig. 4. Control of a robot subsystem using Predictive Control when the reference is a pure step, with the values of the parameters p = 5, m = 3 y λ = 1.

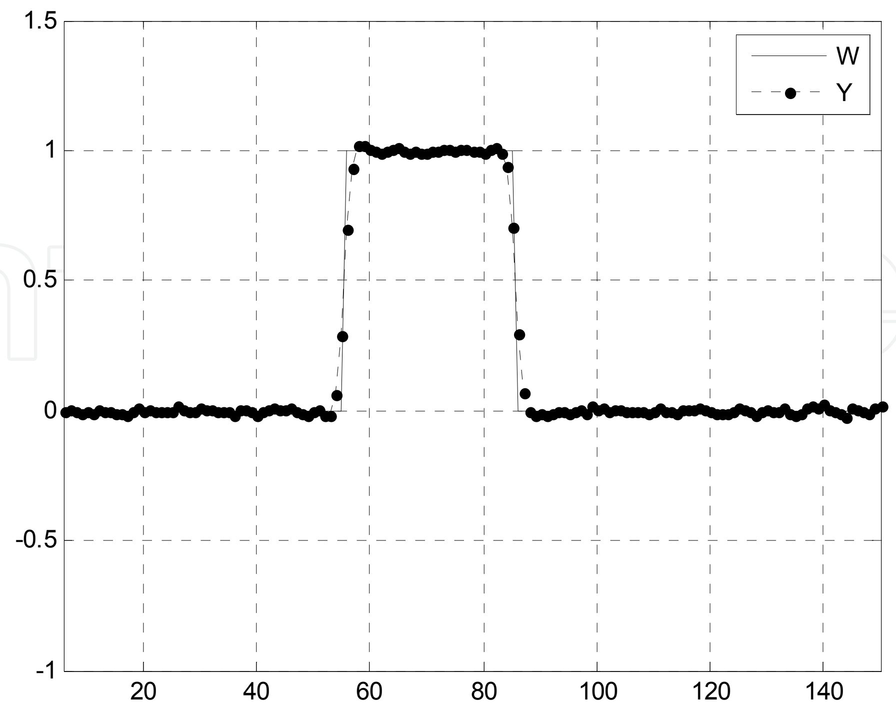


Fig. 5. Control of a robot subsystem using Predictive Control when the reference is a noisy step, with the values of the parameters $p = 5$, $m = 3$ y $\lambda = 1$.

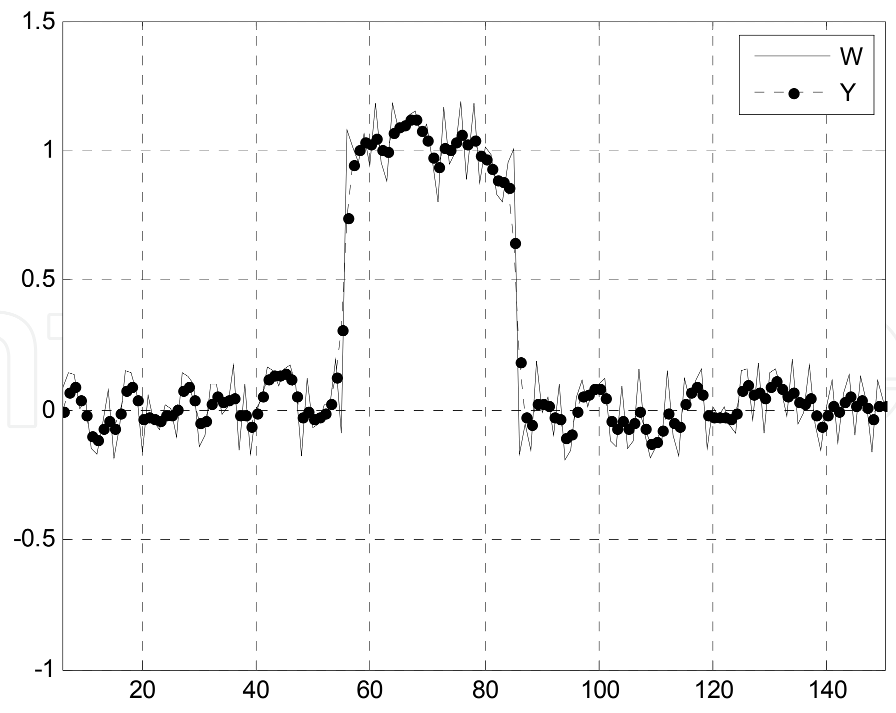


Fig. 6. Control of a robot subsystem using Predictive Control when the reference is a noisy step, with the values of the parameters $p = 5$, $m = 3$ y $\lambda = 1$.

To implement a predictive controller using a neural network we have done training experiments with multiple structures, varying two structural parameters: the number of the hidden layer neurons *h* and the number of delays of the time delay line *d*, having in mind that linear function is computationally efficient.
We have used the Levenberg-Marquardt method to carry out the training of each structure, and the training model has consisted of a target vector $P = \left[w(k), y(k), \Delta u(k-1)\right]'$ and an output $\Delta u(k)$ to get the same control that equation (6).
As it has be shown in Fig. 7, there is a perfect control when we use references that we have used in the training phase of the time delayed neural network. In Fig. 8 and Fig. 9, we can see that the control of the neuronal controller is right even with noisy references that hadn't been used in the training phase.
To implement these predictive controllers using neural networks we have chosen FPGA devices. We have used a device commercialized by Altera Corporation, the EPF10K70 device, in a 240-pin power quad flat pack (RQFP) package.
The way that we have used to implement the neural network in this device is to describe the behavior of that neural network using VHDL languaje, including in the entity that is in this description the same inputs and outputs that the neural network has. VHDL is a description language used to describe the desired behavior of circuits and to automatically synthesize them through specific tools.

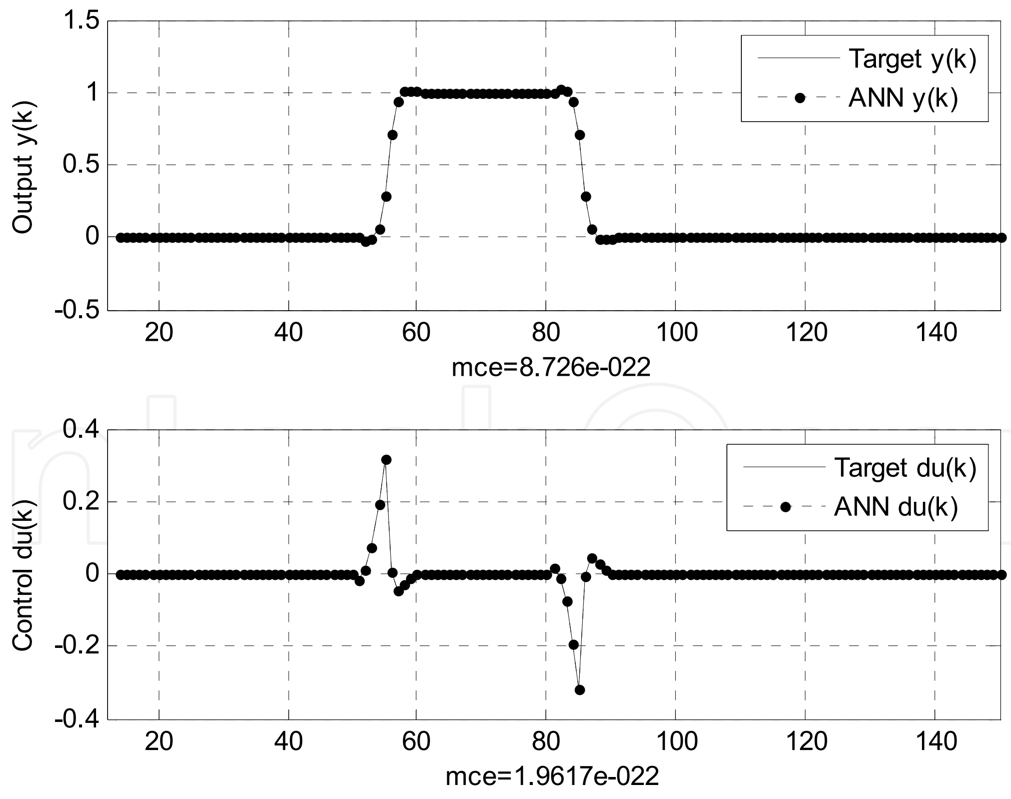


Fig. 7. Control of a system with a Time Delayed Neural Network with a time delay line of *d* = 7 delays in the input, and *h* = 5 neurons in the hidden layer. The reference to follow is a signal that the neural network has been used in the training phase.

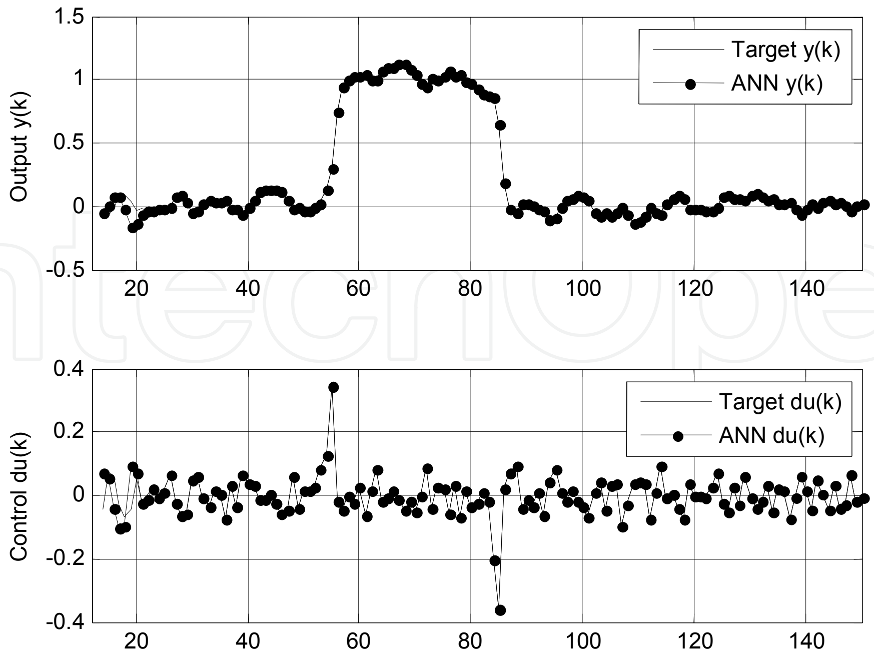


Fig. 8. Control of a robot subsystem with a Time Delayed Neural Network with a time delay line of d = 7 delays in the input, and h = 5 neurons in the hidden layer. The reference to follow is a signal that the neural network hasn't seen in the training phase.

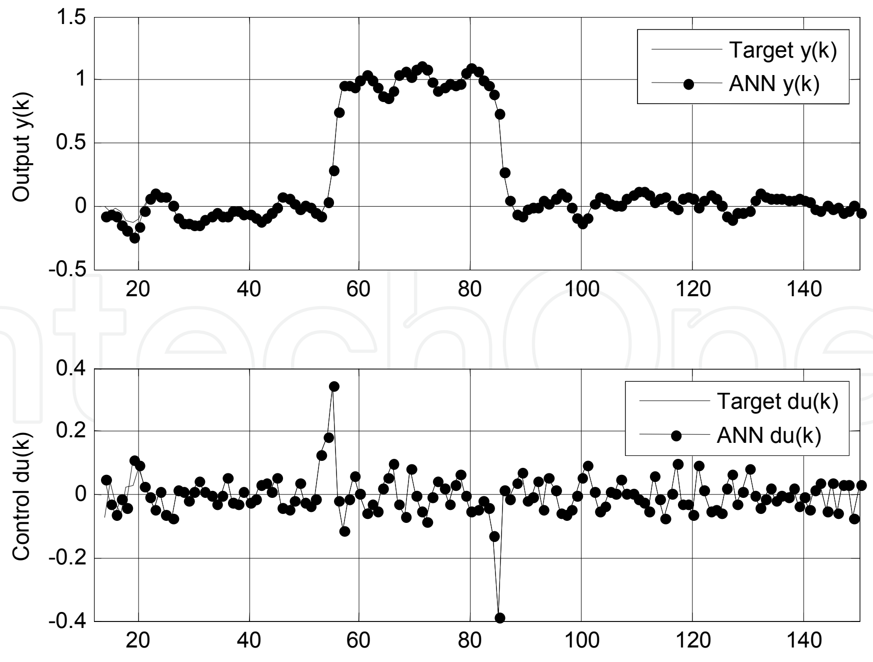


Fig. 9. Control of a robot subsystem with a Time Delayed Neural Network with a time delay line of d = 7 delays in the input, and h = 5 neurons in the hidden layer. The reference to follow is a signal that the neural network hasn't seen in the training phase.

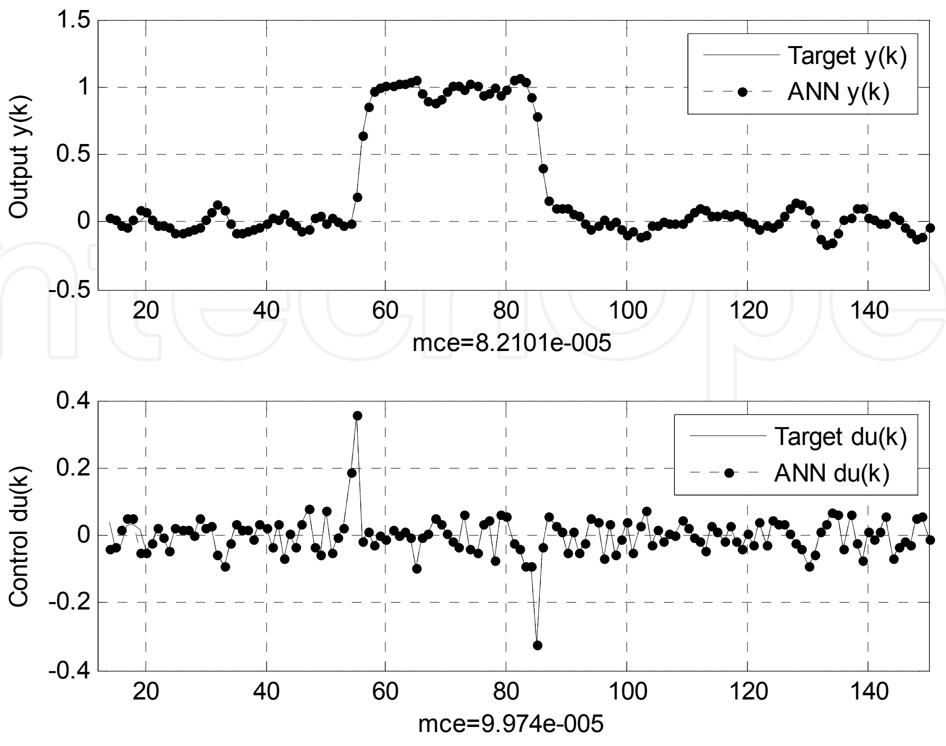


Fig. 10. Control of a robot system with a Time Delayed Neural Network with a time delay line of d = 7 delays in the input, and h = 5 neurons in the hidden layer. The reference to follow is a signal that the neural network hasn't been used in the training phase.

5. Conclusions

This paper has started thinking about the convenience that the computational capacity of robots that belong to multi-robot systems was devoted exclusively to high level functions they have to perform due to being a member of such system. However, each robot must have so many internal control loops as subsystems, and in some cases they aren't controllable through classic techniques. In these cases, predictive control is a good option because it can deal with subsystems that classical PID controllers can't, but it's computationally expensive. In this paper it has been shown how the predictive controllers can be modeled using Time Delayed Neural Networks, which implementation is very cheap using very low cost FPGAs. This way we can reduce de price of each member of multi-robot system, because the investment in computational capacity must cover only the high level functions, ignoring the subsystems that it had, which are solved with very low cost FPGAs.

6. References

Aleksic, M., Luebke, T., Heckenkamp, J., Gawenda, M., et al. (2008). Implementation of an Artificial Neural Network to Predict Shunt Necessity in Carotid Surgery. *Annals if Vascular Surgery*, 22, 5, 635--642.

Andrews, B. W., Passino K. M., Waite, T. A. (2007). Social Foraging Theory for Robust Multiagent System Desing. *IEEE Transactions on Automation Science and Engineering*, 4, 1, 79--86

Arahal, M.R., Berenguel, M., Camacho, E.F. (1998). Neural identification applied to predictive control of a solar plant. *Control Engineering Practice*, 6, 333--344

Braspenning, P. J., Thuijsman, F., Weijters, A. (1995*). Artificial neuronal networks. An introduction to ANN theory an practice*. Springer-Verlag, Berlin.

Camacho, E.F., Bordons, C. (2004). *Model Predictive Control*. Springer-Verlag, London.

Camacho, E. F., Bordons, C. (1995). *Model Predictive Control in the Process Industry*. Springer-Verlag, London.

Chester, M. (1993). *Neural Networks*. Prentice Hall, New Jersey.

Huang, J. Q., Lewis, F. L., Liu, K. (2000). A Neural Net Predictive Control for Telerobots with Time Delay. *Journal of Intelligent and Robotic Systems*, 29, 1--25

Huang, B.Q., Rashid, T., Kechadi, M.T. (2006). Multi-Context Recurrent Neural Network for Time Series Applications. *International Journal of Computational Intelligence*. Vol. 3 Number 1, ISSN pp 45--54.

Kang, H. (1991). A neural network based identification-control paradigm via adaptative prediction. *Proceedings of the 30th IEEE Conference on Decision and Control*, 3, 2939-2941.

Kornienko, S., Kornienko, O., Levi, P. (2005). Minimalistic approach towards communication and perception in microrobotic swarms. *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2228--2234

López-Guede, J.M., Graña, M., Zulueta, E., Barambones, O. (2008). Economical Implementation of Control Loops for Multi-robot Systems. *15th International Conference, ICONIP 2008,* Aucland, New Zealand. ISBN: 978-3-642-02489-4

Maciejowski, J. M. (2002). *Predictive Control with Constraints*. Prentice Hall, London.

McKinstry, J.L., Edelman, G.M., Krichmar, J.L. (2006). A cerebellar model for predictive motor control tested in a brain-based device. *Proceedings of the National Academy of Sciences of The United States of America*, 103, 9, 3387--3392

Narendra, K. S., Parthasarathy, K. (1990). Identification and Control of Dynamical Systems Using Neural Networks. *IEEE Trans. Neural Networks*, Vol. 1, NO.1, pp 4--27

Norgaard M., Ravn, O., Poulsen, N.K., Hansen, L.K. (2003*). Neural Networks for Modelling and Control of Dynamic Systems*. Springer-Verlag, London.

O'Hara, K.J., Balch, T.R. (2007). Pervasive Sensor-less networks for cooperative multi-robot tasks. *7th International Symposium on Distributed Autonomous Robotic Systems* (DARS 2004), 305--314

Rawlings, J.B. (1999). Tutorial: Model Predictive Control Technology. *Proceedings of the American Control Conference* San Diego, California. pp 662--676.

Soeterboek, R. (1992). *Predictive control. A unified approach*. Prentice Hall.

Stancliff, S.B., Dolan, J.M., Trebi-Ollennu, A. (2006). Mission Reliability Estimation for Multirobot Team Design. *IEEE International Conference on Intelligent Robots and Systems*, 2206--2211

Sunan, H., Kok T., Tong L. (2002). *Applied Predictive Control*. Springer-Verlag. London.

Taskaya-Temizel, T., Casey, M.C. (2005). *Configuration of Neural Networks for the Analysis of Seasonal Time Series*. LNCS, vol. 3686, pp. 297--304. Springer, Heidelberg

Voicu , M., Lazär, C., Schönberger, F., Pästravanu, O., Ifrim, S. (1995). Predictive Control vs. PID Control of Thermal Treatment Processes. *Control Engineering Solution: a Practical Approach*. 163--174.

Waibel, A., Hanazawa, T., Hinton, G., Shikano, K., Lang, K. (1989). Phoneme Recognition Using Time Delay Neural Networks. *IEEE Transactions on Accoustics, Speech and Signal Processing*, 37:328—339.

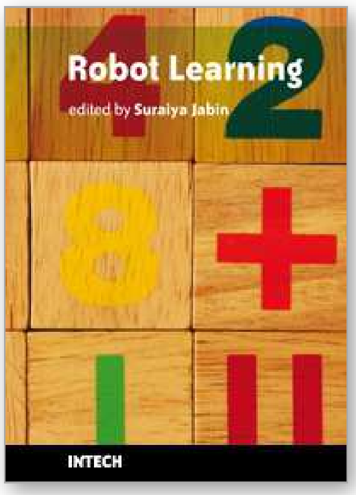
Wang, Y., Kim, S.-P., Principe, J. C. (2005). Comparison of TDNN training algorithms in brain machine interfaces. *Proceedings of the IEEE International Joint Conference on Neural Networks (IJCNN '05),* vol. 4, pp. 2459--2462

Widrow, B., Lehr, M.A. (1990). 30 Years of Adaptive Neural Networks: Perceptron, Madaline, and Backpropagation. *Proceedings of the IEEE*, Vol. 78, No.9. pp 1415 --1442.

Wilson, W.H. (1995). Stability of Learning in Classes of Recurrent and Feedforward Networks. *Proceedings of the Sixth Australian Conference on Neuronal Networks, (ACNN´95)* 142--145.

Wu, H., Tian, G., Huang, B. (2008). Multi-robot collaborative localization methods based on Wireless Sensor Network. *IEEE International Conference on Automation and Logistics*, 2053--2058

IntechOpen



Robot Learning

Edited by Suraiya Jabin

ISBN 978-953-307-104-6

Hard cover, 150 pages

Publisher Sciyo

Published online 12, August, 2010

Published in print edition August, 2010

Robot Learning is intended for one term advanced Machine Learning courses taken by students from different computer science research disciplines. This text has all the features of a renowned best selling text. It gives a focused introduction to the primary themes in a Robot learning course and demonstrates the relevance and practicality of various Machine Learning algorithms to a wide variety of real-world applications from evolutionary techniques to reinforcement learning, classification, control, uncertainty and many other important fields. Salient features: - Comprehensive coverage of Evolutionary Techniques, Reinforcement Learning and Uncertainty. - Precise mathematical language used without excessive formalism and abstraction. - Included applications demonstrate the utility of the subject in terms of real-world problems. - A separate chapter on Anticipatory-mechanisms-of-human-sensory-motor-coordination and biped locomotion. - Collection of most recent research on Robot Learning.

How to reference

In order to correctly reference this scholarly work, feel free to copy and paste the following:

Jose Manuel Lopez-Guede, Ekaitz Zulueta, Borja Fernández and Manuel Grana (2010). Multi-Robot Systems Control Implementation, Robot Learning, Suraiya Jabin (Ed.), ISBN: 978-953-307-104-6, InTech, Available from: http://www.intechopen.com/books/robot-learning/multi-robot-systems-control-implementation

INTECH
open science | open minds

InTech Europe
University Campus STeP Ri
Slavka Krautzeka 83/A
51000 Rijeka, Croatia
Phone: +385 (51) 770 447
Fax: +385 (51) 686 166
www.intechopen.com

InTech China
Unit 405, Office Block, Hotel Equatorial Shanghai
No.65, Yan An Road (West), Shanghai, 200040, China
中国上海市延安西路65号上海国际贵都大饭店办公楼405单元
Phone: +86-21-62489820
Fax: +86-21-62489821

© 2010 The Author(s). Licensee IntechOpen. This chapter is distributed under the terms of the Creative Commons Attribution-NonCommercial-ShareAlike-3.0 License, which permits use, distribution and reproduction for non-commercial purposes, provided the original is properly cited and derivative works building on this content are distributed under the same license.

IntechOpen

IntechOpen