

# Supervising the Reinforcement Learning of a joint control policy

Diego E. Pardo, Cecilio Angulo, Ricardo Tellez  
UPC - Technical University of Catalonia,  
Avda. Victor Balaguer s/n. Vilanova i la Geltrú, Spain.

## Abstract

Along this paper, we propose to model the learning process of the controller policy of a humanoid joint using an already known model of cognition. A policy gradient reinforcement learning method is used as optimization alternative to compute the function that commands the behavior of a robot's motor. The objective is to model the trial-by-trial evolution of the learning and be able to observe its behavior, setting a starting point for the stability analysis of the learning algorithm. In order to achieve it the problem has been split in two input/output models related by a basis function. Dynamical system theory is used as alternative to create a cognition model, instead of the traditional computational hypothesis where cognitive adaptive agents are represented as a collection of logical instructions; this work focuses on the agent environment interaction and the bodily and sensorimotor mechanisms underlying cognitive processes. A general model is presented and a linear implementation of practical use formulated. As a demonstration, the PID control policy of a simulated humanoid's joint is learned and their evolution modeled. The analysis of the results is developed from control system theory and machine learning points of view.

**Keywords** : Reinforcement Learning, adaptive system, cognitive model, sensorimotor control.

## 1 Introduction

Cognitive science has two major approaches. For one side, researchers in this area try to model cognitive processes of the human mind using mathematical tools, allowing to understand knowledge internal procedures of the human brain. The second approach uses the study of cognitive systems for purposes of engineering, seeking to create artificial systems that reliably exhibit some desired level of cognitive performance or behavior. Here we propose a third alternative: Assume that the current machine learning algorithms are artificial cognitive process, thus a model of it can be obtained. By doing this the adaptive systems reveals additional characteristics and therefore a deeper (or at least different) analysis can be performed.

Despite the success and proliferation of the use of learning algorithms in robotic control, the adoption of these algorithms has been held back due to its lack of safety [6], stability conditions can not be established a priori while the controllers are being computed i.e., during learning process. Many stability proofs have been developed for specific online learning techniques, but restrictions are immediately imposed to the complexity of the learning method. Here we change that perspective and show an alternative that fits in the Artificial Intelligence philosophy : The learning algorithms can be modeled as cognitive process and then stability conditions can be implemented very inside of it, without modify its principles and original motivations.

Section 2 overviews the type of cognitive models, and points the importance of the dynamic system as an alternative to handle cognition. Section 3 formulates the cognitive model for learning algorithms that compute a joint controller, then, section 4 shows the use of the model in a specific Reinforcement

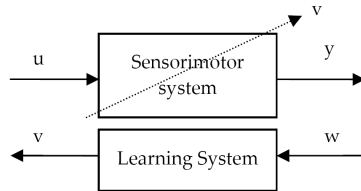


Figure 1: Dynamic Sensorimotor Learning Model

Learning algorithm. Next, the method is implemented during the learning of the controller of a simulated humanoid joint. Finally, conclusions are presented in section 6

## 2 Cognition Models

### 2.1 Overview

During the 60-70s, cognitive science and specially AI, grew up under the influence of an interpretation of the computer metaphor for mind. Therefore, traditional approaches to modeling cognitive systems are computational, using the standard tools and concepts of computation theory. However, recently a number of researchers have argued that cognition is too ‘subtle’ or ‘complex’ for these tools to handle, proposing as an alternative, dynamical systems theory. Opposed to the dominant *computational hypothesis* claiming that cognitive agents are digital computers, the *dynamical hypothesis* claims that cognitive agents are dynamical systems and they can be understood dynamically. Hence, much of the nowadays work in embodied cognitive science is in some sense returning to its cybernetic roots, focusing on agent-environment interaction and the bodily and sensorimotor mechanisms underlying cognitive processes [1].

### 2.2 Modeling Human learning Process

Sensorimotor learning can be defined as an adaptive change in motor behavior in response to sensory inputs. The work presented by Cheng and Sabes [2] defends the dynamical hypothesis, by modeling the learning of a human-sensorimotor task as a linear dynamical system (LDS). They model the *sensorimotor transformation*,  $F(\cdot)$ , as the input-output mapping between sensory signals and motor outputs, and the *sensorimotor learning*,  $L(\cdot)$ , as the algorithm that directs the adaptation of that mapping. Their idea of represent the output of a movement  $y$ , the input-output mapping of the sensorimotor system  $F$ , and the learning process  $L$  is summarized by Eq. (1).

$$\begin{aligned} v_{t+1} &= L(v_t, w_t, \eta_t) \\ y_t &= F(v_t, u_t, \gamma_t) \end{aligned} \quad (1)$$

The parameter vector  $v$  represents the state of the learning and links both models, as it has been depicted in Fig.1. They concentrated in the search of the parameters for the linear model used to represent the functions  $L$  and  $F$ . Since this model was originally conceived to characterize human-like motor learning, very few of the parameters in both, sensorimotor system and learning process were considered known a priori, therefore identification techniques were applied.

The system represented in (1) looks like a standard canonical input - states - output  $(w, v, y)$  model, implying that the approximation made in [2] assumes that the sensorimotor transformation  $F$  is not a dynamic system by itself. Nevertheless, and from a robotic control point of view, the original parameterized model must be modified adding the internal dynamics of the mechanisms being controlled, since the sensorimotor process could be, for instance, a multiple joint robot configuration during a complex (even simple) task, where it is impossible to depreciate the internal dynamics of it.

### 3 General Formulation

The value of the model presented in [2] is that the trial-by-trial evolution of the learning can be supervised. Here we are interested in modeling a mechanical motor system controlled by an adaptive policy. Therefore, we have split the problem in two input-output models: one for the dynamics of learning process and the other one for the dynamics of the controlled plant. These models are linked by a basis function.

#### 3.1 Focusing in the Model of the Learning Process.

Following the ideas of [2], a machine learning process can be modeled using :

$$\begin{aligned}v_{\tau+1} &= L(v_{\tau}, w_{\tau}) \\s_{\tau} &= J(v_{\tau}, w_{\tau})\end{aligned}\tag{2}$$

In the above discrete time dynamical system, the parameter vector  $v_{\tau}$  represent the internal state of the learning at any step  $\tau$ . The states evolve through trials, guided by an algorithm  $L$  which also depends on an input vector  $w_{\tau}$ , that drives the learning. The output of the learning  $s$  depends on the state  $v_{\tau}$  and the input vector  $w_{\tau}$ . Usually this late contains information about the motor plant  $(x, u)$  and exogenous data  $r$ . The input of the adaptation algorithm is formalized using the following function,

$$w = \phi(x, u, r)\tag{3}$$

The purpose of the new defined output variable  $s_{\tau}$  is to use it as a vector parameter of the basis function:

$$h_{\tau}(x, t) = h(x, t, s_{\tau})\tag{4}$$

This basis function will be used as a control policy for the motor system, therefore, it connects the learning dynamical process with the motor control task; this also explains why it contains variables of both.

From the learning process point of view,  $h$  is a parametric function, and the output of (2) fills these parameters. In the other hand, and since the controlled process point of view,  $h$  is used as a controller, i.e, a function that calculates the control action to be applied on it. This signal depends on the internal states of the plant  $x$  and the time process. Note that variables  $x, t$  are not part of (2), they belong to the controlled plant universe, so they do not affect the parameters update process at all.

#### 3.2 Facing the Motor System.

The system to be controlled is the dynamical continuous process described in (5),

$$\begin{aligned}\dot{x} &= f(x, u) \\y &= g(x, u, \zeta)\end{aligned}\tag{5}$$

This standard description includes the states of the plant  $x$ , the inputs  $u$  and a noise term  $\zeta$  which represents the combined effect of sensory, motor and processing variability. As mentioned in the previous subsection, the system would be controlled by an adaptive policy, letting ,

$$u(t) = h_\tau(x, t) \tag{6}$$

The framework defined by Equations (2) to (6) provides a general cognitive model of a sensorimotor system controlled by an adaptive structure.

### 3.3 About the Time-Scale Of the Learning Model.

Parameter  $\tau$  represents the trial step of the learning algorithm, and it is assumed that  $L$  does not explicitly depend on time. This stationary assumption allows linking its model with the one representing the controlled plant.

Every sampling time  $\tau$ , a new function  $h_\tau(x, t)$  will be used as a controller in the motor plant. A straightforward question is *when to take a new step*  $\tau + 1$ ?

Two time-frameworks are considered: Whether to calculate new parameters for the basis function during the control loop process, either to wait until a final result when the settling or maximum process time would be reached. The first represents the adaptive control point of view, in which every single time step of the dynamical process the controller is changed [5],[6], the second one represents a framework in which a learning algorithm is trying to find a controller for the motor focusing in the accomplishment of an specific task [3],[4].

Many well known structures can be fitted in (6), like a simple gain , PID, RBFs, MLP, among others; also many popular search strategies can be fitted in (2). Notice that the structure used to control the plant is described by (4), and the algorithm that finds the parameters of such structure is modeled by (2) and (3). An example has been depicted in Fig (??) for illustration purposes.

## 4 Cognitive Model of an Adaptive PID

The previous description is too general to be of practical use, therefore is necessary to determine the equations of the motor and the methodology employed to learn the controller.

### 4.1 The Control Problem

The task at hands consists in to take the states of a simulated simple-robot system to a desired state ( $x^*$ ) without any ‘pre established’ restriction on the path taken by them to achieve it, i.e., tracking a path is not the task, but reaching a final goal point. A ‘black box’ simulink model of the robot arm is used as a plant, which lineal equations can be written down :

$$\begin{aligned} x_p(t+1) &= A_p x(t) + B_p u(t) \\ y &= C_p x(t) \end{aligned}$$

The states variables intended to be controlled, position and velocity  $x_p = [\theta, \dot{\theta}]$  , are supposed to be directly readable ( $C_p = I$ ) and not calculated using the input ( $D_p = 0$ ), besides  $B_p \in \mathfrak{R}^2$  and  $u_p \in \mathfrak{R}$ . As a controller a conventional discrete PID is employed,

$$u(t) = u(t-1) + K_p e(t) + K_I \sum_{j=1}^t e(j) + K_D [e(t) - e(t-1)]$$

Where  $e$  represents the error vector of the state variables, the state space representation of the PID is given by,

$$\begin{aligned}x_c(t+1) &= A_c x_c(t) + B_c \Delta u(t) \\ \Delta u(t) &= C_c x_c(t) + D_c e(t)\end{aligned}$$

The composite closed loop state space system can be written as:

$$\begin{bmatrix} x_p(t+1) \\ x_c(t+1) \end{bmatrix} = A_d \begin{bmatrix} x_p(t) \\ x_c(t) \end{bmatrix} + \begin{bmatrix} B_c D_c \\ B_c \end{bmatrix} e(t) \quad (7)$$

Elements of  $A_d$  are factor which contain  $A_p, B_p, C_p$  and  $A_c, B_c, C_c, D_c$ , this last containing the PID gains  $\kappa = [K_P K_I K_D]$ .

## 4.2 Adaption Law

In order to measure the quality of the set of parameters selected to the controller  $\kappa(\tau)$ , lets consider the cost function  $J$  to minimized as,

$$J(\tau) = \frac{1}{2} [x(\infty)^* - x(\infty)]^2 = \frac{1}{2} e^2(\infty) \quad (8)$$

Every time step  $\tau$ , the algorithm updates the set of gains by the actualization rule,

$$\theta(\tau) = \theta(\tau - 1) + \eta g(\tau) \quad (9)$$

where the vector  $g(\tau)$  represents a search direction and the positive scalar  $\eta$  is the learning rate, which determines the convergence speed. It is hoped that the cost functional  $J(\tau)$  decreases at each iteration,

$$J(\tau + 1) < J(\tau) \quad (10)$$

To ensure this last condition, the update made in (9) should be,

$$g(\tau) = \nabla J(\theta) \quad (11)$$

To compute (11) the following chain rule is employed:

$$\frac{\partial J}{\partial \theta_i} = \sum_j \frac{\partial J}{\partial x_j} \cdot \frac{\partial x_j}{\partial \theta_i} \quad (12)$$

It means that every learning step  $\tau$  a new vector of parameters is calculated as a linear combination of the previous ones plus an input  $g$ , which include the change in  $J$  respect to  $x(\infty)$ ,

$$\frac{\partial J}{\partial x(\infty)} = -e(\infty) \quad (13)$$

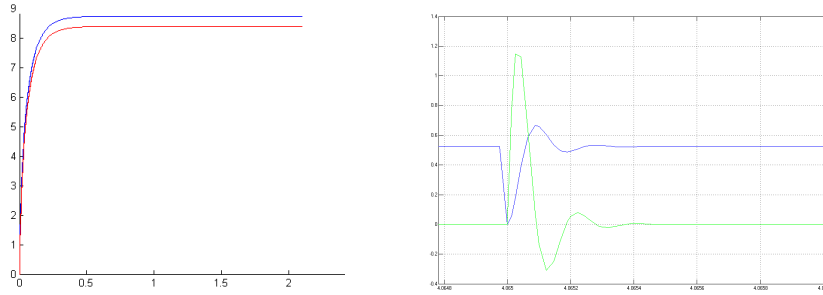


Figure 2: Evolution of the learning states. & Robot output response to a step input.

### 4.3 Learning The PID

Once the model of both process have been established, a trial by trial interaction between the simulated joint is proceed in order to approximate the unknown factors of Eq.(11). The action-perception process empathize the cognitive motivation of the learning.

**About the Time Framework.** As described in section 2, the update of the parameters of (7) is performed at the end of the task, i.e,  $\tau$  will increase after the controller is tested. Therefore, it can be easily view that the value of  $g_\tau$  is proportional to the last stationary error of the robot signals. Based on this fact, the external input  $r_i$  is interpreted as the desired final state of the position and velocity for the robot arm. Two types of external input were used for  $\theta$ , a step and a ramp, in both cases the desired velocity was zero. i.e, positioning the robot arm.

## 5 Results.

### 5.1 Outputs.

Fig 2. shows the position and velocity behavior of the robot link using a fully learned controller. The task of positioning the link in the desired value ( $\theta = 30^\circ$ ) with zero final velocity is accomplished. This figure corresponds to a trial step  $\tau = 100$  , in which the parameters of (7) have converged. Fig 2. shows the evolution of the states ,  $v$  , during the learning procedure. It can be observed how stable states are reached, i.e., the task has been learned.

A similar simulation has also been performed using a ramp ( $slope = 1$ ) as a reference. Even when a ramp is inserted as input, the learning finds a controller that drives the robot to the desire state. This example shows how the '*task-oriented*' framework differs from the '*path following*' one. The behavior of the position signal through time is not a  $45^\circ$  ramp like the reference, but it accomplished the 'task' imposed to it.

## 6 Conclusions

Much of the nowadays work in embodied cognitive science is in some sense returning to its cybernetic roots, focusing on agent-environment interaction and the bodily and sensorimotor mechanisms underlying cognitive processes. Here we are interested in modeling a mechanical motor system controlled by a learning algorithm; therefore the problem has been split in two dual input-output models related by a basis function. The general formulation presented has been initially constrained to the all linear case by using a linear learning function (the delta rule). The performed simulation on a one-link arm robot has allowed relating the observed results with those of the use of a PID control structure. Only initial results has been obtained on simulation, however the easy learning rule used will allows us to theorize about learning, adaptation and, in this early stage, study its relation with PID control.

## 7 Work in Progress

Based on the previous analysis, learning schemes to find controllers that exploit the dynamics of multiple joint robots are being studied. Humanoids, with multiple degrees of freedom and redundancies, developing coordinated 3-D tasks are the target of our current efforts.

### References

- [1] T. Ziemke (2005) "Cybernetics and embodied cognition: on the construction of realities in organisms and robots", *Kybernetes*, 34(1/2):118-128.
- [2] S. Cheng, P.N. Sabes (2006) "Modeling Sensorimotor Learning with Linear Dynamical Systems", *Neural Computation*, 18(4):760-793.
- [3] M.T. Rosenstein, A.G. Barto (2001) "Robot Weightlifting by Direct Policy Search", in *Proceedings of the Seventeenth International Joint Conference on Artificial Intelligence*, Bernhard Nebel, Ed., San Francisco, 2001, vol. 2, pp. 839–844, Morgan Kaufmann.
- [4] R. Téllez, C. Angulo, D. Pardo (2005) "Highly Modular Architecture for the General Control of Autonomous Robots", *Lecture Notes in Computer Science*, vol. 3512, pp. 709-716 (IWANN'05 - Vilanova i la Geltrú, Spain), Springer.
- [5] Y. Diao, K.M. Passino (2004) "Stable adaptive control of feedback linearizable time-varying non-linear systems with application to fault-tolerant engine control" *International Journal of Control*, vol. 77, pp. 1463-1480, Taylor& Francis.
- [6] A.Y. Ng, H.J. Kim (2005) "Stable Adaptive control with online learning", *Advances in Neural Information Processing Systems*, Lawrence K. Saul, Yair Weiss, Léon Bottou Eds., Cambridge, MA, vol. 17, pp. 977-984, MIT Press.
- [7] G. Widrow, M.E. Hoff (1960) "Adaptive switching circuits", *Institute of Radio Engineers, Western Electronic Show and Convention, Convention Record*, 4:96–104.
- [8] Y.Li, K.H Ang, G.C.Y Chong (2006) "PID Control Systems Analysis and Design", *IEEE Control Systems Magazine*, 26 :32-41
- [9] A.O'Dwyer (2003), *Handbook of PI and PID controller Tuning Rules*. London: Imperial College Press.
- [10] R.M.Lewis, V.Torczon, M.W. Trosset (2000) "Direct Search Methods:Then and now", *Journal of Computational and Applied Mathematics*, 124:191-207