Proceedings of the
44th IEEE Conference on Decision and Control, and
the European Control Conference 2005
Seville, Spain, December 12-15, 2005

Macro-Actions in Model-Free Intelligent Control with Application to pH Control

S. Syafiie, F. Tadeo, and E. Martinez

***Abstract* – MFIC (Model-Free Intelligent Control) is a technique, based on Reinforcement Learning, previously proposed by the authors to control processes without needing a precalculated model. In standard reinforcement learning algorithms (including MFIC), the interaction between an agent and the environment is based on a fixed time scale: during learning, the agent can select several primitive actions depending on the system state. This creates the problem of selecting a suitable fixed time scale to select control actions, to trade off accuracy in control against learning complexity and flexibility. A novel solution to this problem is presented in this paper: Macro-actions, that incorporate a general closed-loop policy and temporal extended actions. The application of macro actions on a laboratory plant of pH process shows that the proposed MFIC learns to control adequately the neutralization process, with reduced computational effort.**

I. INTRODUCTION

Reinforcement Learning (RL) algorithms are based on online learning directly from the closed-loop behavior of the system. The main idea behind these algorithms is '*learning what to do by doing'*, i.e. how to map perceptions of process states to control actions, so as to maximize an externally provided scalar reward signal.

In standard RL frameworks, a learning agent interacts with an environment at some discrete time scale (t=1,2,3,etc). At each time step, t, the environment is in some state, s_t. In current state, s_t, the agent selects an action, a_t, and executes it, the environment responses to the action and presents to the agent the state transition, s_{t+1}, and the reward, r_{t+1} (See Figure 1). State transitions depend on the preceding state and action, also may depend on it in a stochastic fashion. The mapping from state to action (called p*olicy*) is to learn to maximize the discounted return (a weighted sum of rewards). Details of the algorithms were proposed in [1] and an application by the authors to pH process was first presented in [2].

Manuscript received September 15, 2005. This work was supported by MCYT-CICYT (DPI2004-07444-C04-02). The first author would like to thank the MAE-AECI for financial support.

S. Syafiie and F. Tadeo are with the Department of System Engineering and Automatic Control, Faculty of Sciences, University of Valladolid, Prado de la Magdalena s/n, 47011 Valladolid, Spain (email: {syam;fernando}@autom.uva.es).

E. Martinez is with the Consejo Nacional de Investigaciones Cientificas y Technicas, Avellaneda 3657 3000, Santa Fe, Argentina (email: ecmarti@ceride.gov.ar).

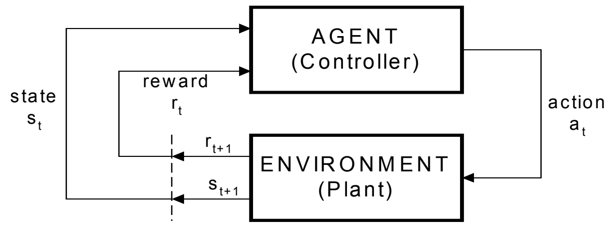


Fig. 1. Components of reinforcement learning algorithms

Although RL Algorithms has been proved successfully in areas like robotics or games, standard reinforcement learning algorithms, like Q-learning, are not simple to implement in process control problems, as shown by the small number of published applications in this area [3,4,5]:

- First, it is necessary to transform adequately the control requirements to the definition of states, actions and rewards, for the different problems found in Process Control (command tracking, disturbance rejection), taking into account issues like robustness or reduced control efforts that are specific of Process Control problems. In previous works [2,6] the authors proposed simple methods to carry out this transformation: the algorithm MFIC, based on the control structure in Figure 2, which will be used in this paper, which is to introduce *extended* actions.
- Second, these algorithms scale very badly with increasing problem size, granularity of states or control actions. One intuitive reason for this, among others, is that the number of decisions from the initial state to the goal state increases exponentially with the number of states and actions. Unfortunately for most real process control problems the number of possible states and actions is rather big, which makes it difficult to implement these algorithms using inexpensive hardware. This paper proposes to solve this problem by using *Macro-actions*, as will be described later.

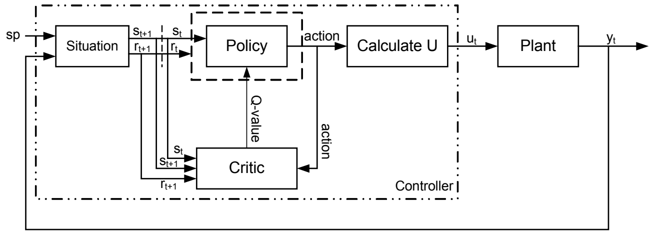


Fig. 2. MFIC architecture

Compared to other control techniques based on learning, the reinforcement learning approach to model-free control design has some clear advantages:

- It can optimize control signal over choosing action during the online interactions between an agent and an *environment*.
- It is possible to put in the design of the controller previous knowledge of the system
- The control algorithm is quite simple from a computational point of view, so it is feasible to implement using low-cost hardware.
- It is possible to derive and obtain a feedback control law from an optimal control point of view based on actual experience rather than a process model, which makes it attractive for Control and Plant Engineers.

According to the problem size, to keep tractable the number of decision to be taken to reach the goal state, hierarchical approaches based on *temporal abstraction* have been proposed. Temporal abstraction can be defined as an explicit representation of *extended actions*, as policies together with a termination condition [7]. The original one-step action is called *primitive action*. Semi Markov Decision Processes (SMDPs) is the theory used to deal with temporal abstraction as a minimal extension of reinforcement learning framework. SMDPs is a Markov Decision Processes (MDP) appropriate for modeling continuous-time discrete-event systems.

Several reinforcement learning algorithms resorting to hierarchical temporal abstraction approaches have been recently proposed: Hierarchy of Abstract Machine (HAM) [8]; MaxQ [9] and Multi-step actions (MSA) [10]. The first two methods are based on the notion that the whole task is decomposed into subtasks each of which corresponds to a subgoal. MSA is a method that the agent learns to implement multiple-fixed-time-scale, for example for *m*-time-step termination condition [6, 10]. The *macro-actions* use in this paper are similar to the temporal extended actions proposed on [11], where it was called *Options*. Options may be either multiple step policies or primitive actions while macro-actions are restricted to temporally extended actions [12].

The macro-actions enable a learning agent to learn a control policy by using multiple time scales until the system reaches termination conditions. Those, we think for applying the macro-actions algorithms to allow the agent learns the environment more flexibility and speed up learning and planning to achieve the goal of the controller. This makes possible to apply MFIC to process control problems where the number of states might be big and simplify its implementation in microcontrollers, to simplify its implementation in industrial process control problems.

II. MACRO-ACTIONS

In this paper the concept of macro actions [12] is applied to process control because it is suited for systems where no decomposition in subproblems is known in advance. As in the general framework defined on [11], macro-actions are a special type of semi-Markov option. A Markov option would require a state-dependent termination condition. Macro-actions algorithm is a closed loop policy with termination conditions. When the macro-actions completes a new primitive or macro-actions can be selected. Macro-actions are often use in robotics; macro-operators are also well known as an aid to state-space search in artificial intelligent systems.

The macro-actions can be chosen at the same level as primitive actions. Macro-actions commit the learning agent to act in particular, purposeful way for a sustained period of time. Also, macro-actions may either accelerate or retard learning, depending on the appropriateness of the macro-actions to the particular task [13]. The macro-actions method is a method enabling an intelligent control to learn a control policy by using multiple time scales until the system reaches termination conditions.

Action in macro-actions is extended in time for more than one time step. This could be to reduce the number of states of underlying MDP to states connected by macro-actions. Therefore, application of macro-actions in complex problem can speed learning and training compared to a nonhierarchical system and can obtain solution to harder problem than can be solved using only primitive action. Due to the agent selects macro actions and complete it when the system reaches termination conditions, the macro-actions algorithm is possible to increase responsiveness and add flexibility to the controller behavior. Thus, we think that the algorithm can be extended to complex and highly nonlinear problem, such as pH control problem.

The MFIC based on macro-actions can address many of weaknesses inherent in traditional PID or other advanced control methods. For example PID is basically linear and time-invariant and cannot effectively control complex processes that are nonlinear, time variant, coupled, and have large time delays, major disturbance and uncertainties. While, model predictive control is designed using empirical model of the plant, so this controller cannot be used for general acid-base systems because each model only represents the specific process.

III. *Q*-LEARNING WITH MACRO-ACTIONS

This paper proposes the application of the macro-actions algorithm for process control. The advantages of MFIC based on macro-action are the flexibility the agent selects the optimal action and performs the control signal, no precise quantitative knowledge of the process is available, no process identification mechanism (identifier is included in the system, which is an online learning), no controlled design for a specific process is needed, simple manual tuning of controller parameter is required and stability analysis criteria are available to guarantee the closed-loop system stability.

Macro-actions are policies with termination conditions. Each macro-action is specified by a closed-loop policy, which determines the primitive action when the macro actions are in force, and by a completion function, which determines when the macro-action ends. On each time step, the agent can choose either a macro-action or a primitive action, unless it is already executing a macro-action. Once the agent has chosen a macro-action, it selects the primitive actions in accordance with the macro-action's policy until the macro-action's termination condition is satisfied.

To provide for learning when to select macro-actions, the notion of optimal action-value function is extended to Q^*, to include macro-actions. This extended action-value function can be defined as $Q^*(s, m)$ for each state s and macro-action m, as the maximal expected return given that the agent start macro-actions m in state s. This definition naturally leads to update rule: upon each termination of a macro-action, its value is updated using the cumulative discounted reward received while executing the macro-action and the maximum value at the resulting state. More precisely, after a multi-step transition from state s_t to state s_n using macro action m, the approximate action value $Q(s_n, m)$ is updated by:

$$Q(s_t,m) \leftarrow Q(s_t,m) + \alpha\left[r + \gamma^n \max_{a \in A} Q(s_n,a) - Q(s_t,m)\right] \qquad (1.)$$

where the max is taken over both action and macro-actions, α is positive step size parameter and the total reward is calculated as follows:

$$r = r_{t+1} + \gamma r_{t+2} + \cdots + \gamma^{n-1} r_{t+n}$$

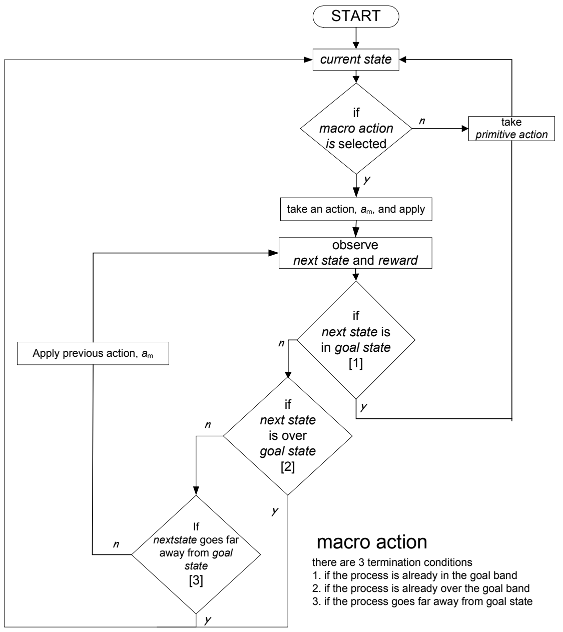
where γ is a discount factor that increases the effect of short-term rewards with respect to long-term ones.

Then, the basic MFIC algorithm with macro actions is as follows:

1. Read the state s_t
2. Select either primitive or macro-actions
3. If macro action is selected apply the selected one until termination condition
4. Update Q-value using Equation (1)

Compared to standard reinforcement learning, this modified Q-learning algorithm is proposed for process control problems because it can extract more training examples from the same number of experiments (that in the proposed macro-actions approach consists on applying an action until the termination condition is reached). If the macro actions are selected, the agent executes a primitive action and applies it for multiple time steps until termination conditions.

Reinforcement learning based on macro-actions seems to be a promising approach to process control problems because the control law can easily adapt to varying scenarios by online learning. By applying a sequence of actions or macro actions, the agent can speed up learning and planning to maintain the process in the desired pH value. In order to explore the set of possible actions and acquire experience through the reinforcement signals, the actions are selected using an exploration/exploitation policy. In this study *ε-greedy policy* is applied to select one of the available actions in visited state and experience it for a multiple time steps of



START
current state
if macro action is selected
n
take primitive action
y
take an action, a_m, and apply
observe next state and reward
if next state is in goal state [1]
Apply previous action, a_m
n
y
if next state is over goal state [2]
n
y
If nextstate goes far away from goal state [3]
n
y
macro action
there are 3 termination conditions
1. if the process is already in the goal band
2. if the process is already over the goal band
3. if the process goes far away from goal state

Fig. 3. Implementation of macro action and definition of termination conditions.

the plant. The ε-greedy policy has been selected because, based on previous experiments, it gives better performance for pH process than *softmax* policy [2].

The termination condition for this application is where the process reaches the goal band (condition 1) or pass the goal band (condition 2) or goes far away from the goal state (condition 3). The idea is shown in Figure 3. Therefore, in this application there are 3 termination conditions introduced.

This macro action can be understood as following, for example, when the process is in state 4, the agent can either to select macro actions or primitive action. It is introduced 10% probability to select macro actions. If the agent selects macro actions, the agent will execute it until termination conditions are reached. The idea is if the system is in state 4 the agent selects macro actions (according to macro actions, the agent selects a primitive action) and makes transition to the next state and receive reward. If next state is goal state (condition 1) the macro actions are terminated or if the next state is over goal state (condition 2), for example the next state is state 7, the macro actions are terminated, or if the next state goes far away from goal state (condition 3), for example next state is state 3, the macro actions are terminated. Whereas, if the next state is neither goal state nor pass goal state, nor far away from goal state, the agent continues executing macro actions, which is a previous action that the agent has chosen.

Due to a simple state action definition, this paper is to check the feasibility of application macro-actions to a simple defined problem, that previously it has been applied to pH process [2,6]. Of course this small simple problem presented in this paper is not really necessary to use macro actions unless definition of state and action are growth very big. Even thought, this proposed method can be extended to a big state action definition of pH processes in order to speed up learning and planning. For higher dimensional problem with plenty of states and/or actions, macro actions will be necessary to implement RL.

IV. APPLICATION TO A NEUTRALIZATION PROCESS

This section describes the implementation of the macro-actions approach on a pH neutralization process. This problem was selected, because control of pH in neutralization processes is a ubiquitous problem encountered in chemical and biotechnological industries. In most pH neutralization processes the control of pH is not only a control problem but also the chemical equilibrium, kinetic, thermodynamic and mixing problems must also be considered [14]. These characteristics make it difficult to control pH process. Another problem is the process buffer capacity, which is unknown and dramatically changes process gain. It could be difficult for control design.

Also, due to the nonlinear dependence of the pH value on the amount of titrated agent the process will be inherently nonlinear. Moreover, variations of the buffering effects could make the process time-varying. Therefore, it is difficult to develop an appropriate mathematical model of the pH process for designing proper controller. The unique pH process characteristic is difficult to control the process even using complex linear or nonlinear system. All of these make the pH process difficult to control using classical process control techniques [15].

Several control strategies have been previously applied for neutralization processes; for instance, Fuzzy Control [16], Fuzzy Internal Model Control [17], Fuzzy Predictive Control [18], and Neural Networks [15]. Unfortunately, in these approaches there are some weaknesses, such as:

- o complexity of the control structures (which could be difficult to implement on existing control systems),
- o conservativeness (the controllers take a long time to reject disturbances and to reach the desired setpoint),
- o difficulty of tuning, which makes it a time-consuming task (these controllers have many tuning parameters, or require many experiments before its application to a real industrial process).

A. States and Reward

The most important decision when applying MFIC is to give an adequate characterization of the states in the system ("situation" in Figure 4). For pH control problems, in a previous work [6] it was shown that selecting the states based solely on the main measured output (pH) was enough to get adequate performance. Of course for more complex control systems it might be necessary to include additional information to define the states: other outputs, the control signals, etc. However this definition of states makes possible to derive simple termination conditions for macro-actions, as it is shown later.

As the main control objective is to maintain the pH inside a band of $\pm\delta$ around the desired setpoint (the width of this band is defined by measurement noise in the process and allowed tolerance), this band is defined as the *goal* state. The rest of the states were defined corresponding to values of the pH outside this band, as depicted in Figure 1.

In this work, 11 states where selected, where the goal state is 6 (corresponds to the desired pH band). Each state has 5 possible actions, except in the goal state that has only 1 action, which is called *wait action* (that corresponds to keeping the control signal constant). This number of states was selected purely by experience [6].

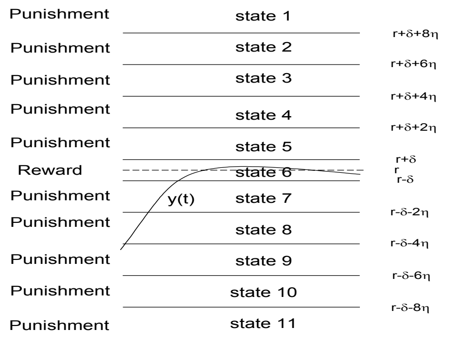


Fig. 4. Control objective and definition of states.

The probability that the system moves to a new state from the current state depends on the system behavior following the execution of the chosen action.

To define the reward, only positive reward is given if the system is in the goal state. That is, the reward function is:

$$reward = \begin{cases} 1 & if\ r-\delta < pH < r+\delta, \\ -1 & else \end{cases} \qquad (2.)$$

B. Control Actions

As in all reinforcement learning algorithm, once the agent has selected an action (based on the Policy, see Figure 2), it must be applied to the system and receive the next reward, so that the Critic can evaluate the action. For process control problems this means that the action selected must be transformed into a control signal (The “Calculate U” block in Figure 2). In MFIC the control signal, u_t, is calculated incrementing (or decrementing) the previous control signal by an amount calculated from the difference from the selected action to the wait action. That is,

$$u_t = u_{t-1} + k\left(a_w - a_t\right) \tag{3.}$$

where a_t is the optimal action chosen by the agent from those available actions in visited state, and a_w is *wait action* where there is no variation of the previous control signal. The controller gain, k, is a tuning parameter that can be selected to weigh how much to increase or decrease previous control signal over the action chosen (that can be tuned by the operator as in classical process control) .

V. EXPERIMENTAL RESULTS AND DISCUSSION

The experimental results section describes and discusses the application of MFIC of macro actions approach based on reinforcement learning to control a pH process laboratory plant.

A. *Description of the Experimental Setup*

The experimental setup consists of a continuous stirred tank reactor (CSTR) where a process stream (sodium acetate) is titrated with solution of hydrochloric acid (HCl), and the effluent is to be maintained at certain pH value by manipulation titrating flow.

The control variable u_t is the flowrate of the titrating stream (normalized to the maximum value). The output variable, y_t, is the logarithmic hydrogen ion concentration (pH) in the reactor. It is assumed that the mixing is homogeneous, therefore the concentration in the effluent stream is similar to the concentration in the reactor. The plant is controlled and monitored from a personal computer, using Matlab and the Real-Time Toolbox for online control.

B. *Parameters Selection*

For the experimental process, the zero initialized Q-value is used. The value of the *meta – parameter* for the agent are selected to be: discount factor, γ=0.98 and learning rate, α=0.1, which were determined to be a good values from previous work [2].

In MFIC, the ε-greedy policy is applied for choosing an action in every visited state of the pH process. Parameter ε used in the ε-greedy policy is selected to be ε=0.1. The wait action is chosen to be 22, which is no manipulation of previous control signal. The gain of MFIC is chosen to be 2x10^{-5}. This gain gave good performances [2].

C. *Experimental Results and Discussion*

Application of the proposed MFIC based on macro-actions to the laboratory plant shows good results. The responses of the plant for some changes in setpoint controller can be seen in Figure 5 for the sodium acetate –

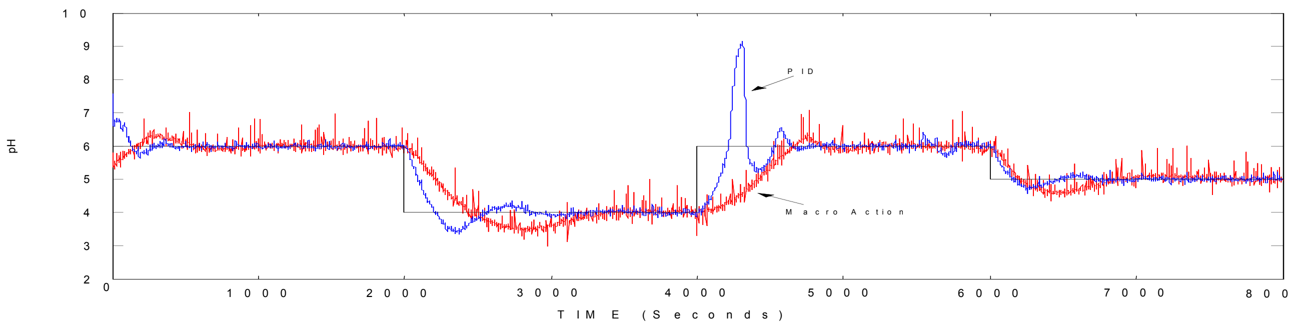


Fig. 5. Output Responses of the plant for NaCH3COO-HCl systems.

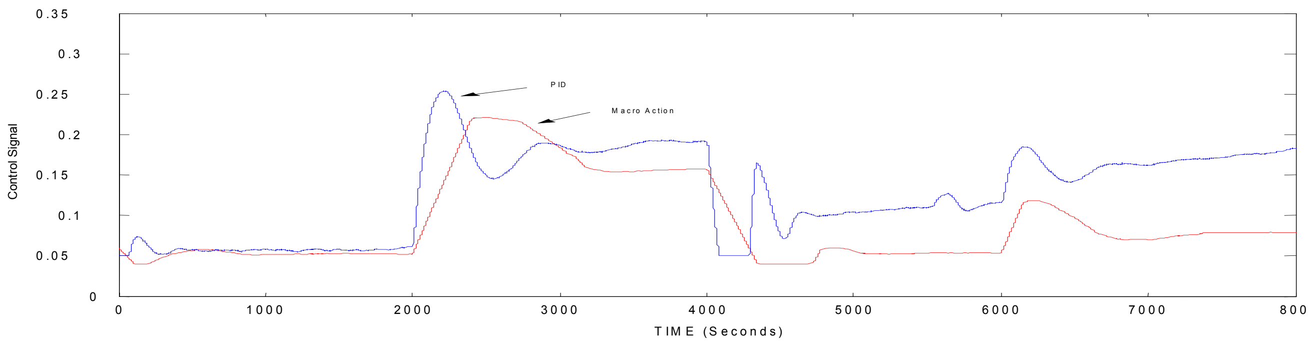


Fig. 6. Control signal for NaCH3COO-HCl systems.

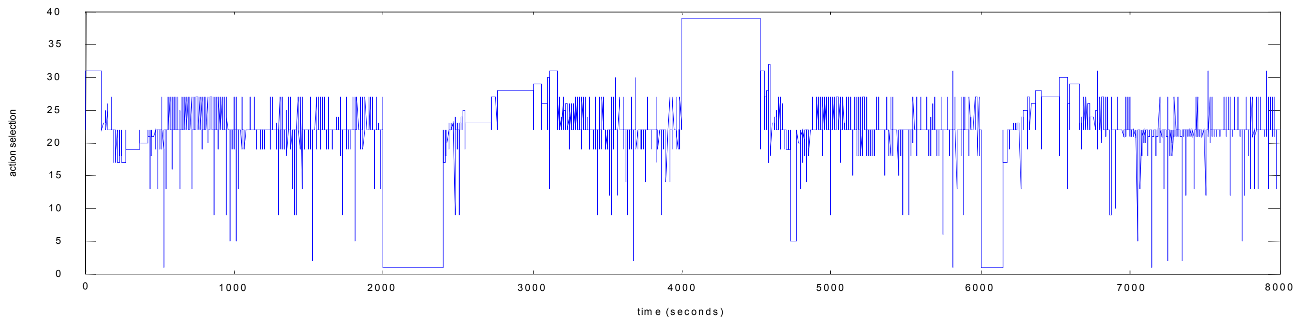


Fig. 7. Action selections during learning.

hydrochloride acid system. The responses of the plant show that the proposed MFIC of macro-actions controller based on reinforcement learning algorithms lay close to the references and reach the reference after the environment change than the responses using PID controller. This PID controller was tuned based on operating condition at pH=5, where correction gain and proportional gain are chosen 0.01 and 0.001 respectively.

The control signal (Figure 6) shows that the proposed MFIC of reinforcement learning based on macro-actions controller manipulates the actuator smoothly. Since MFIC allows a tolerance error of the process whenever the pH is within the control band, the control signal is smoother when the process is closer or within the pH band.

To check the improvement of exploration time, it is possible to see in Figure 7, for example at time 2000. During 800 sample time, for standard RL, every sample time the agent learns to explores *n* actions and executes them. Whereas using this proposed macro-actions the agent does not explore at all, these 800 sample times, as there is a clear selection for the control signal. From this, it is clear that the controller exploration time is greatly reduced. Also, Figure 7 clearly shows that the agent is either to select a primitive action or a macro actions more flexible.

VI. Conclusions

A modification of the Model-Free Intelligent Control (MFIC) algorithm has been proposed, based on macro-actions. This makes possible to reduce computational effort, as in most sample times, it is only necessary to check the termination condition, making it feasible to implement in inexpensive hardware.

The technique has been applied on a pH control problem at laboratory scale, for sodium acetate – hydrochloride acid. It has been shown that the behavior of the pH control with application of macro-actions gives good performance, with reduced computational effort. It is noteworthy the smoothness of the resulting control signal. Thus, the proposed technique is promising for process control problems.

References

[1] Sutton R. S., and A. G. Barto (1998), *Reinforcement Learning: an Introduction*, the MIT press, Cambridge, MA

[2] Syafiie S., F. Tadeo, and E. Martinez (2004), Softmax and ε-Greedy Policies Applied To Process Control, *IFAC Workshop on Adaptive and Learning Systems (ALCOSP04)*, Yokohama, Japan, August 2004, pp.729 – 734.

[3] Riedmiller M. (1998), High quality thermostat control by reinforcement learning-A case study, in *Proceedings of the Conald Workshop* 1998, Carnegie Mellon University.

[4] Tchamitchian M., C. Kittas, T. Bartzanas, C. Lykas (2005), Daily temperature optimisation in greenhouse by reinforcement learning, 16th *IFAC World Congress*, Prague, Czech Republic, July 2005

[5] Martinez E. C. (2000), Batch process modelling for optimization using reinforcement learning, Computer and chemical engineering, Vol. 24, pp. 1187 – 1193.

[6] Syafiie S., F. Tadeo, and E. Martinez (2005), Model-Free Intelligent Control Using Reinforcement Learning and Temporal Abstraction-Applied to pH control, 16th *IFAC World Congress*, Prague, Czech Republic, July 2005.

[7] Precup D. (2000), *Temporal Abstraction in Reinforcement Learning*, Ph.D. Dissertation, Department of Computer Science, University of Massachusetts, Amherst.

[8] Parr R. (1998), *Hierarchical Control and learning for Markov decision processes*, PhD thesis, University of California at Berkeley

[9] Dietterich T. G. (1997) Hierarchical Reinforcement Learning with the MAXQ Value Function Decomposition, *Technical report*, Department of Computer Science, Oregon State University.

[10] Riedmiller M. (1998), High quality thermostat control by reinforcement learning-A case study, in *Proceedings of the Conald Workshop* 1998, Carnegie Mellon University.

[11] Sutton R.S., Precup, D., Singh, S. (1999). Between MDPs and semi-MDPs: A Framework for Temporal Abstraction in Reinforcement Learning. *Artificial Intelligence,* **Vol. 112**, No.1 –2, pp.181-211.

[12] McGovern A., and R. S. Sutton (1998), Macro-Actions in Reinforcement Learning: An Empirical Analysis, *Amberst Technical Report Number 98-70.*

[13] McGovern A., R. S. Sutton and A. H. Fagg (1997), Roles of Macro-Actions in Accelerating Reinforcement Learning, *proceeding of the 1997 Grace Hopper Celebration of Women in Computing.*

[14] Gustafsson T.K., Skrifvars B. O., Sandström K.V., Waller K. V. (1995), Modeling of pH for Control, *Industrial Engineering Chemical Research*, **Vol. 34**, pp. 820-827.

[15] Loh, A. P., K. O. Looi, and K. F. Fong (1995), Neural network modeling and control strategies for a pH process, *Journal of Process Control*, **Vol. 6**, pp. 355 – 362.

[16] Fuente M. J., C. Robles, O. Casado, S. Syafiie, and F. Tadeo (2002), Fuzzy control of neutralization process, IEEE CCA 2002, Glasgow.

[17] Edgar, C. R., and B. E. Postlethwaite (2000), MIMO fuzzy internal model control, *Automatica*, **Vol. 34,** pp. 867 – 877.

[18] Biasizzo, K. K., I. Skrjanc, and D. Matko (1997), Fuzzy predictive control of highly nonlinearity pH process, *Computers and Chemical Engineering*, **Vol. 21,** pp. s613 – s618