15th Triennial World Congress, Barcelona, Spain

INTELLIGENT PROCESS SUPERVISION USING RENFORCEMENT LEARNING AND TEMPORAL ABSTRACTION

Ernesto Martinez

INGAR- Instituto de Desarrollo y Diseño (CONICET)
Avellaneda 3657, S3002 GJC, Santa Fe, Argentina

Abstract: Supervisory control usually involves timely switching among different courses of action over multiple time scales. In this work, intelligent process supervision is addressed in the context of semi-Markov decision processes and reinforcement learning. Temporally extended actions that represent a way of behaving together with a termination condition are used to achieve a set of operational goals/sub-goals comprising a supervision task. The control strategy resorts to a hierarchy of macro-actions or *options* which are made up of closed-loop sequences of low-level, primitive actions. Supervisory control of a buffer tank is discussed as a representative example. *Copyright © 2002 IFAC*

Keywords: Hybrid Control, Reinforcement Learning, Process Control, Supervisory Control, Semi-Markov Decision Processes.

1. INTRODUCTION

Most of the surveillance functions (including start-up and shut-down operations, safeguarding, real-time optimization and throughput changeover) of complex process plants involving multiple objectives and nonlinear, time varying dynamic characteristics are still in the realm of human control (Samad and Cofer, 2001). This type of control tasks are generally poorly structured due to its *behavioral nature*, that is the aim of control actions is to guarantee a given process functionality or achieving operational, safety and economic targets that cannot merely be expressed as regulation or tracking activities. Also, there often exist tight energy and material coupling within the overall process give rising to plant-wide effects of each control action which slowly unfold over time. As a result, the degree/type of automation that is being used for process supervision is very limited (Lind, 1999).

Process supervision is a complex, ill-defined control function (Peterson, 2000), where information processing and decision making take place at different time scales (Sutton et al, 1999; Precup, 2000). The key issue to be tackled is how a supervision agent might select the “most appropriate” or the “most relevant” course of action to follow at a given time, when facing a particular process state. Because of process dynamic complexity and outcome unpredictability, course of action selection can hardly be expected completely rational, let alone optimal. It is thus more appropriate to search for control strategies that in practice are *efficient* and *sufficiently satisfying*, i.e., “good enough” rather than optimal. Process supervision necessarily calls for planning, reasoning and learning using temporal and state abstraction. In this work, the issue of learning and planning with macro-actions in process supervision is addressed by integrating together reinforcement learning and temporal abstraction. The approach we follow has been adapted from the theory of Semi-Markov Decision Processes (SMDP), which are used to model continuous-time, discrete-event systems (Lunze et al, 2001). Supervisory control of a buffer tank is discussed as a representative example.

2. TEMPORAL ABSTRACTION

A central concern in process supervision is planning, reasoning and decision making so as to achieve operational goals at different time scales by resorting to available process means, i.e. functions and physical components. For instance, consider the operating procedure used by a human operator for starting up a distillation column. Should it reason and plan at the lower level of valve openings and rates of heat input? Or should planning be concerned with high-level actions such as "pressurize the column" or "develop tray temperature profile". For operating procedures to be compact high-level actions and long-term dynamics should be emphasized. However, planning and reasoning at this level alone is not enough to implement the plan or to change it as may be required by unforeseen circumstances. Fine-grained plans allow filling in enough details but are too focus on short-term dynamics to be used efficiently alone. Ideally, a process supervisor should be able to reason and learn at both levels of abstraction in parallel and from the real experience that it gets by acting on the physical components of the system, namely sensors and actuators.

A key issue in temporal abstraction is how best represent using a unified framework low-level actions in the supervision domain along with courses of action that can be temporarily extended and goal-directed. For doing this, a novel concept named "option" has been proposed by Sutton et al. (1999) and Precup (2000). An option is specified by an activation set $\mathcal{I}$, a way of picking actions or internal policy π, and a termination condition β stating when an option should be ended. Ending an option may be because a certain goal/sub-goal has been successfully achieved or switching to a different option seems preferable. The set $\mathcal{I}$ describes system conditions under which a given option can be initiated. The activation set and termination conditions of a macro-action together restrict its range of application in a potentially useful way. Sometimes is useful for a given macro-action to "timeout," that is to terminate after some period of time has elapsed, even if they have failed to achieve the desired behaviour or goal being pursued.

Assuming that the supervision task can be modeled as a Markov Decision Processes at the lower level of process actuators (e.g. valves and set-points), the introduction of macro-actions allows modeling decision-making at the upper level of abstraction as Semi-Markov Decision Processes (SMDP). Formally, a SMDP is a tuple $<\mathcal{S},\mathcal{A},\mathcal{R},\Pi>$, where $\mathcal{S}$ is the continuum of process states, $\mathcal{A}$ is the set of macro-actions, $\mathcal{R}$ is the supervision reward function, and Π is the joint distribution of the next state and transit time. If system is in state s and the macro-action m_i is implemented at the current decision epoch, then $Q\left(t,s'\middle|s,m_i\right)$ denotes the probability that the next decision epoch occurs within time t and that the system will be in state s' at that time. r_s^m is the reward obtained in the transition $s \to s'$.

3. PLANNING AND LEARNING

3.1 SMDP Planning

Planning with macro-actions require a model of their consequences. For any macro-action m, let $\varepsilon(m,s,t)$ denote the event of m being initiated in state $s \in \mathcal{I}$ at time *t*. Then, the reward associated with following the course of action m from is:

$$r_s^m = E\left\{r_{t+1} + \gamma . r_{t+2}, ... + \gamma^{k-1} r_{t+k} \middle| \varepsilon(m,s,t)\right\} \quad (1)$$

where *r* are step-by-step rewards associated with the supervision task, *t*+ *k* is the random time at which m terminates and $\gamma \le 1$ is a *recency* factor to geometrically discount rewards within each macro-action. The discount factor γ is used to weight near term reinforcements more heavily than distant future reinforcements.

The state-prediction part of the model of m for state *s* is:

$$p_{ss'}^m = \sum_{k=1}^{\infty} p(s',k)\gamma^k, \forall s' \in \mathcal{S} \quad (2)$$

For a given *k*, $p(s',k)$ is the probability that the macro-action terminates in s' after *k* decision steps, involving primitive, low-level actions. Thus, $p_{ss'}^m$ is a combination of the likelihood that s' is the state in which the macro-action m is terminated together with a measure of how delayed that outcome is relative to γ. This kind of model is called a multi-time model (Sutton, 1999; Precup, 2000) because it describes state changes at potentially many different times in the sequel.

Multi-time models can be used to write Bellman equations for supervision policies using macro-actions. For any Markov policy μ over macro-actions, the state value function can be written as:

$$V^{\mu}(s) = E\left\{r_{t+1} + \gamma . r_{t+2}, ... + \gamma^k V^{\mu}(s_{t+k}) \middle| \varepsilon(m,s,t)\right\} \quad \ldots(3)$$

Using the multi-time model in equation (2), this equation can re-written as:

$$V^{\mu}(s) = \sum_{m \in \mathcal{A}} \mu(s,m) \left[r_s^m + \sum_{s'} p_{ss'}^m V^{\mu}(s') \right] \quad (4)$$

which is the well-known Bellman equation for the state-value function, whereas $\mathcal{A}$ is the set of macro-actions. The corresponding Bellman equation for the value of initiating a macro-action m in state $s \in \mathcal{I}$:

$$Q^{\mu}(s,m) = r_s^m + \sum_{s'} p_{ss'}^m \sum_{s'} \mu(s',m')\ Q^{\mu}(s',m') \quad \ldots(5)$$

Similarly, the optimal Bellman equations for states and macro-actions are, respectively,

$$V_{\mathcal{A}}^{*}(s)=\max_{m\in\mathcal{A}}\left[r_s^m+\sum_{s'}p_{ss'}^m V_{\mathcal{A}}^{*}(s')\right] \quad (6)$$

and

$$Q_{\mathcal{A}}^{*}(s,m)=r_s^m+\sum_{s'}p_{ss'}^m \max_{m'\in\mathcal{A}_{s'}} Q_{\mathcal{A}}^{*}(s',m') \quad (7)$$

Should $V_{\mathcal{A}}^{*}$ is known and models of the macro-actions are available, then the optimal supervision policy is merely choosing at each state s the macro-action that returns the highest value for the resulting state s'. If $Q_{\mathcal{A}}^{*}$ is known, then the optimal supervision policy is choosing at each state s' the macro-action m that:

$$Q_{\mathcal{A}}^{*}(s,m)=\max_{m'} Q_{\mathcal{A}}^{*}(s,m') \quad (8)$$

3.2 SMDP Learning

The problem of finding an optimal policy over a set of macro actions $\mathcal{A}$ is addressed here along the lines developed earlier Bradke and Duff (1995) and Sutton et al. (1999). In a given state, the learning controller picks a macro-action m and executes it until its termination conditions apply, which causes the state transition $s\to s'$. Based on the experience accumulated between s and s', an approximate value function for macro-actions $Q(s,m)$ is updated. The basic learning equation is the *SMDP Q-learning* update rule that follows each macro-action termination:

$$Q(s,m)\leftarrow Q(s,m)+\alpha\left[r_s^m+\gamma^k \max_{m'\in\mathcal{A}_{s'}} Q_{\mathcal{A}}^{*}(s',m')-Q(s,m)\right] \quad .....(9)$$

here k denotes the number of time steps elapsing between s and s' and α is the learning rate which may depend arbitrarily on the states, macro-action and number of time steps. Since there exists a continuum of process states s, a key issue to apply the *SMDP Q-learning* rule in (9) is safely approximating the value function $Q(s,m)$.

For safe interpolation of the value function while training the supervisory controller we resort here to a query-based prediction algorithm that revolves around the idea of *locally weighted regression* (LWR). LWR is a variation of standard linear regression techniques in which training points that are closer to the query point have more influence over the approximation than those further away (Atkeson *et al.*, 1997). A new regression is performed as required for every query point during learning. This results in a globally nonlinear model while retaining simple, locally linear models that can be easily estimated with well-known statistical techniques.

The SMDP Q-learning algorithm for training an intelligent supervisor is shown in the Fig. 1. Training exemplars are supplied as 4-tuples (s,m,r_s^m,s') that summarize the reward and state transition that result of taken the macro-action m at state s. First the prediction set K containing already visited state-action pairs that neighbour (s,m) is identified. In Step 1 and 2, we estimate both the value $Q(s,m)$ and the maximum value from the resulting state s' using LWR. Step 4 implements the basic update rule in (8) resulting in a new value estimation $Q_{new}(s,m)$. This new value is then stored in the "memory" of the LWR subsystem for future use. Finally, the value of each pair in K is updated by bringing it closer to $Q_{new}(s,m)$ depending on the proximity as measured by its kernel weighting κ_i.

```
Input:
   Experience, $(s,m,r_s^m,s')$
   Learning rate, $\alpha$
   Discount factor, $\gamma$
   LWR bandwidth, $h$
0:  $K=\{k_i\}\leftarrow$ Recall set of exemplars for LWR
1:  $q\leftarrow Q_{predict}(s,m)$ using LWR
2:  $q_{next}\leftarrow \max_{\mathcal{A}} Q_{predict}(s',m')$
3:  $\kappa_i\leftarrow \exp(-(q-k_i)^2/h^2)$
4:  $q_{new}\leftarrow q+\alpha(r_s^m+\gamma\, q_{next}-q)$
5:  Memorize $Q(s,m)\leftarrow q_{new}$
6:  for each pair $(s_i,m_i)$ in $K$ do
       $Q(s_i,m_i)\leftarrow Q(s_i,m_i)+\kappa_i(q_{new}-Q(s_i,m_i))$
```

Fig. 1. *SMDP Q-learning* algorithm using locally weighted regression.

4. SUPERVISORY CONTROL

Buffer tanks are frequently used in the process industry to alleviate the impact downstream of disturbances in temperature, concentration and flow rate in important process streams (Tani *et al.,* 1996). The simplest example is shown in Fig. 2. There exists a process stream having a significant and unsystematic variation in its flow rate that needs to be fed to a heater which cannot accept rapid variations in its inflow rate. To dampen outflow rate variations a buffer tank is used. So, one component of the tank desired behavior is that its outflow rate must be *changed smoothly* despite significant changes in the incoming flow rate on a long-term basis. To satisfy this requirement the level in the tank needs to be constantly varied within its operational minimum and maximum limits. But the tank is having a limited capacity that should be used appropriately. Thus, keeping the tank level *stable* is also an important component of the desired behavior.

What does an intelligent controller need to know or learn so as to achieve successfully the desired tank behavior? The key to success is *anticipation* to assess the long-term impact of following a course of action. For example, if the inflow rate will present a peak in the next hour a gradual increase in the outflow rate will prevent overflowing the tank without affecting the heater operation. On the contrary, if a minimum in the inflow rate is expected, slowly diminishing the outflow rate will permit to build up a tank inventory to avoid stop feeding the heater. In the end, what does really matter is the resulting behavior from a given control strategy, namely will tank outflow rate be smoothly changed throughout? Will the event of tank overflow or tank becoming empty be avoided? Summing up, the supervision task is defined as:

Supervision task: *Manipulating the outflow valve in such a way that the outflow rate is changed smoothly in the face of significant and unsystematic inflow variations, while the tank level does not overflow nor becomes low enough to prevent providing a minimum outflow rate downstream.*

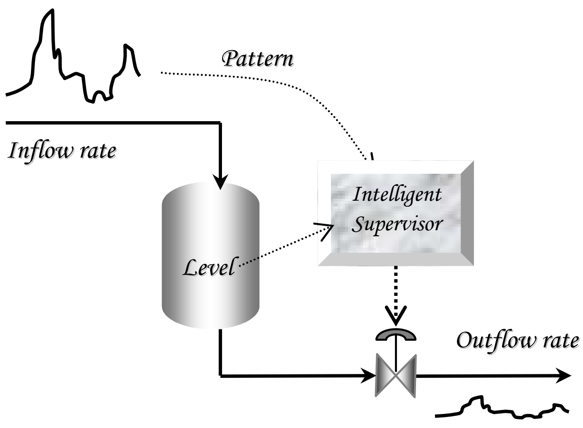


Fig. 2. The buffer tank. An intelligent supervisor manages tank limited capacity to filter downstream inflow variations.

4.2 Conceptual design of macro-actions

Successful supervision of the tank on a continuous basis demands from the intelligent agent to effectively implement a small set of course of actions aimed at achieving qualitative changes in the tank state. Course of actions or macro-actions describe generic behaviors or goals that having different activation sets and termination conditions. However, regardless of these defining elements, the conceptual design of courses of action for a given behavior is the same. For the buffer tank we have four basic behaviors as follows.

Accumulate. The desired behavior in this case is to gradually increase the tank hold-up without introducing an abrupt change in the opening of the outflow valve. Thus, the reward to be received by the agent at time t+1 is defined as

$$r_{t+1} = -\|u(t) - u(t-1)\| \quad \text{if} \quad h(t+1) > h(t)\text{, and}$$
$$r_{t+1} = -1\text{, otherwise.} \qquad (10)$$

where $u_{min} \le u(t) \le u_{max}$ is the opening of the outflow valve scaled to the interval [0,1] and $h_{min} \le h(t) \le h_{max}$ is the tank level, also scaled to [0,1].

Balance. The desired behavior or control goal here is to equilibrate the inflow rate with the outflow rate to avoid significant changes in the tank hold-up. Rewards for control actions are defined so as to maintain level within a certain ξ-band around the tank level $\hbar$ when the option was first initiated:

$$r_{t+1} = -\|u(t) - u(t-1)\| \text{ if } \|h(t+1) - h(t)\| \le \xi\text{, and}$$
$$r_{t+1} = -1\text{, otherwise.} \qquad (11)$$

Drain-off. For this case, control actions pursue to gradually decrease the tank hold-up without introducing an abrupt change in the opening of the outflow valve. The rewards are defined as

$$r_{t+1} = -\|u(t) - u(t-1)\| \quad \text{if} \quad h(t+1) < h(t)\text{, and}$$
$$r_{t+1} = -1\text{, otherwise.} \qquad (12)$$

Wait. This macro-action corresponds to maintaining the outflow rate at its current level for a certain interval of time. Since no changes in valve opening are made, rewards are constant throughout and equal to the maximum value of *zero* while this option is implemented.

Assuming that an inductive time-series model $\mathrm{I}f_{t+1} = \varphi(\mathrm{I}f_t, \mathrm{I}f_{t-1}, \mathrm{I}f_{t-2}...)$ is available for short-term predictions of the inflow rate upon plant data records, the implementation of a given macro-action can be conveniently cast as the following optimization problem:

P-macro: given the current tank state $s = \{u(t), h(t)\}$, along with a known sequence of previous inflow rates, solve:

$$\max_{\Delta u(t+1),...\Delta u(t+n)} \hat{r}_s^m \qquad (13)$$

Subject to:

$$u_{min} \le u(t+j) \le u_{max}\text{, } j = 0,1,...,n \qquad (13.1)$$
$$h(t+j+1) = tank(\varphi(\bullet), u(t+j), h(t+j)) \qquad (13.2)$$

Eqn. (13.1) in *P-macro* is used to enforce the manipulated variable bounds on the valve opening $u(t)$, while Eqn. (13.2) is the hold-up mass conservation model for the tank that describes the (simulated) effect of control actions according to the predictions of the inflow rate to the tank. To allow a gradual opening or closing of the outflow valve, each macro-action is internally achieved with a cycle time of 30 seconds and the moving prediction horizon was chosen so that n=24. The MATLAB function for constrained optimization "`constr`" was used to determine a profile of valve changes that minimize the predicted reward for the macro-action $\hat{r}_s^m$.

4.2 Inflow rate modeling and prediction

For the buffer tank in Fig. 2, let's assume that the inflow rate varies in a pseudo chaotic way described by the Mackey-Glass differential delay equation (Mackey and Glass, 1977)

$$\frac{d\mathrm{I}f}{dt} = \frac{0.2\mathrm{I}f(t-\tau)}{1+\mathrm{I}f^{10}(t-\tau)} - 0.1\,\mathrm{I}f(t), \quad \left[\frac{\textbf{litres}}{\textbf{min}}\times 10^{-2}\right] \quad \ldots(14)$$

with $\tau = 17$ and $\mathrm{I}f(0) = 1.20$. The solution of Eqn. (12) provides a non-periodic and non-convergent time series that is very sensitive to initial conditions.

The goal of the predictive model is to use past values of the inflow time series up to time t to predict the inflow rate entering the tank at some point in the future $t + P$. The standard approach to carry out this prediction is to fit a mapping from D previous data points of the inflow rate time series spaced Δ apart- that is, $[\mathrm{I}f(t-(D-1)\Delta),\ldots,\mathrm{I}f(t-\Delta),\mathrm{I}f(t)]$, to a predicted future value $\mathrm{I}f(t+P)$. For the present study, the values $D = 4$ and $\Delta = P = 6$ have been adopted. Accordingly, input-output data pairs of the form:

$$[\mathrm{I}f(t-18),\mathrm{I}f(t-12),\mathrm{I}f(t-6),\mathrm{I}f(t),\mathrm{I}f(t+6)]$$

have been used to fit a classical time-series model with exponential forgetting.

4.3 Intelligent supervision

To train an intelligent supervisor for the tank, activation sets and termination conditions in Table 1 and Table 2, respectively, are used by the *SMDP Q-learning* algorithm. Note that we can meaningful expand the set of macro-actions simply by choosing different terminating conditions. A larger set of macro-actions might allow a more elaborated behavior of the intelligent controller. However, this will depend on the prior knowledge at hand to establish meaningful terminating conditions.

The following process perception has been chosen so as to introduce a certain amount of "memory" for learning:

$$s_t = [h(t), u(t), \mathrm{I}f(t), \mathrm{I}f(t-6), \mathrm{I}f(t-12), \mathrm{I}f(t-18)]$$

Each training episode consists of implementing a succession of macro-actions that satisfy their activation sets (Table 1) and terminating conditions (Table 2), according to their current values, provide the maximum cumulative reward until the tank either overflow or becomes empty. Episodes differ in the tank initial holdup (high) that is chosen randomly between [10-90]%, whereas the inflow rate time series is continued further from the previous episode. The evolution over training of the squared Belman error E_B:

$$E_B = \left[r_S^m + \gamma^k \max_{m' \in \mathcal{A}_{s'}} Q_{\mathcal{A}}^*(s', m') - Q(s, m) \right]^2 \qquad (16)$$

corresponding to all those states for which terminating conditions in Table 2 apply, is shown in Fig. 3. For a cautious learning rate of $\alpha = 0.1$, in the order of 5,000 episodes are required to learn the value Q of implementing each macro-action depending on the tank perception s_t. The controlled operation of the tank using the trained intelligent supervisor over a 4-hours period is shown in Fig. 4.

5. CONCLUDING REMARKS

The intelligent process supervision approach presented here highlights the benefits of temporal abstraction in the design of options or macro-actions. Timely switching among different courses of action over multiple time scales permits to commit an intelligent supervisor to act in a purposeful, useful way for long periods of time. Current research is aimed at providing a logical foundation for specifying and implementing intelligent supervision agents based on the integration between reinforcement leaning and temporal abstraction.

REFERENCES

Atkenson, C. G., A. W. Moore and S. Schaal (1997). Locally weighted learning, *Artificial Intelligence Review*, **11**, pp. 11.

Bradke, S. J. and M. O. Duff (1995), Reinforcement learning methods for continuous time Markov decision processes, in proceedings of *Advances in Neural Information Processing Systems*, G. Tesauro, D. Touretzky and T. Leen (eds.), **7**, 393, MIT Press, Cambridge, MA, 1995

Lind, M. (1999). Plant modeling for human supervisory control, *Trans. Inst. Measurement and Control,* **21**, pp. 171

Lunze, J., B. Nixdorf, B. and H. Ritcher (2001). Process Supervision by means of a hybrid model, *J. of Process Control*, **11**, pp. 89.

Mackey M. C. and L. Glass (1977). Oscillations and chaos in physiological control systems, *Science*, **197**, pp. 287.

Peterson, J., (2000). *Knowledge-based support for situation asssessment in human supervisory control*, Ph. D. Thesis, Chapter 1, Department of Automation, Technical University of Denmark.

Precup, D. (2000). *Temporal abstraction in reinforcement learning*. Ph. D. Thesis, University of Massachusetts, Amherst, MA.

Samad, T. and D. Cofer (2001). Autonomy in Automation: trends, technologies and tools, *Computer-aided Chemical Engineering*, **9**, pp. 1.

Sutton, R. S., D. Precup and S. Singh, (1999). Between MDPs and semi-MDPs: a framework for temporal abstraction in reinforcement learning, *Artificial Intelligence,* **112**, pp. 181.

Sutton R. S. and A. G. Barto (1998). *Reinforcement Learning: An Introduction*, The MIT Press, Cambridge, MA.

Tani, T., S. Murakoshi and M. Umano (1996). Neuro-fuzzy hybrid control system of tank level in petroleum plant, *IEEE Transactions on Fuzzy Systems,* **4**, 360.

Table 1. Activation sets for macro-actions

Behavior	*Activation Set*
Fast Accumulation	$\{h(t_0) \le 0.1 \wedge u(t_0) > 0.1\}$
Accumulation	$\{h(t_0) \le 0.5 \wedge u(t_0) > u_{\min}\}$
Balance	$\{0.10 \le h(t_0) \le 0.90 \wedge u_{\min} \le u \le u_{\max}\}$
Wait	$\{h_{\min} \le h(t_0) \le h_{\max} \wedge u_{\min} \le u \le u_{\max}\}$
Drain-off	$\{h(t_0) \ge 0.5 \wedge u(t_0) < u_{\max}\}$
Fast Drain-off	$\{h(t_0) \ge 0.9 \wedge u(t_0)] < 0.90\}$

Table 2. Terminating conditions for macro-actions#

Behavior	*Terminating condition*
Fast Accumulation	$\beta(j) \le 0.01, \forall j > 10 \quad or \;\; u(t) < 0.1 \forall t > t_0$
Accumulation	$\beta(j) \le j \times 0.001, \forall j > 1 \;\; or \;\; u(t) < 0.05 \forall t > t_0$
Balance	$\beta(j) < -0.01 \quad or \, \beta(j) > 0.01, \forall j > 10$
Wait	$\beta(j) < -0.05 \quad or \; \beta(j) > 0.05, \forall j > 10$
Drain-off	$-\beta(j) \le j \times 0.001, \forall j > 1 \;\; or \;\; u(t) > 0.95 \, \forall t > t_0$
Fast Drain-off	$\beta(j) \ge -0.01, \forall j > 10 \;\; or \quad u(t)] > 0.90 \;\; \forall t > t_0$

$$^{\#}\,\beta(j) = \sum_{t=t_0}^{t=t_0+j} \left[h(t+1) - h(t_0)\right]$$

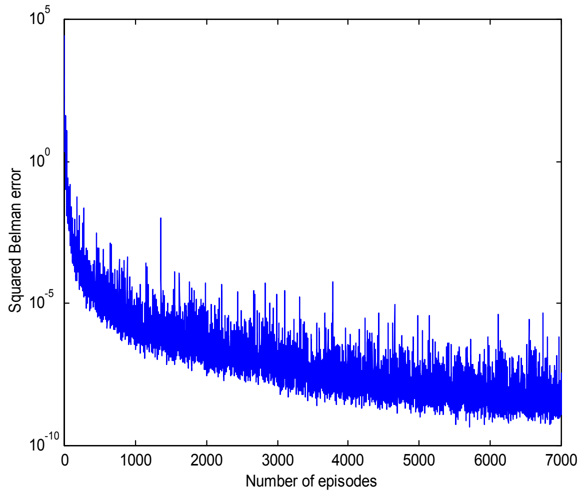
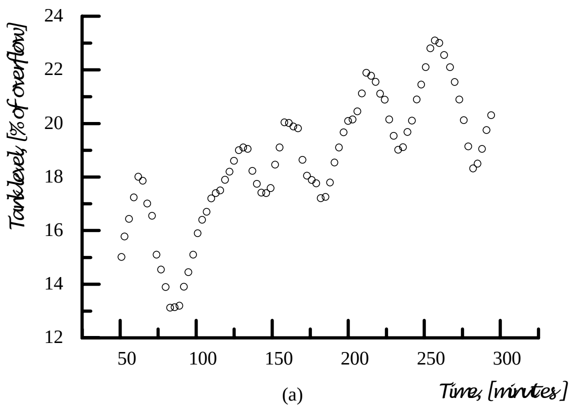


Fig. 3. Learning curve for the supervision agent



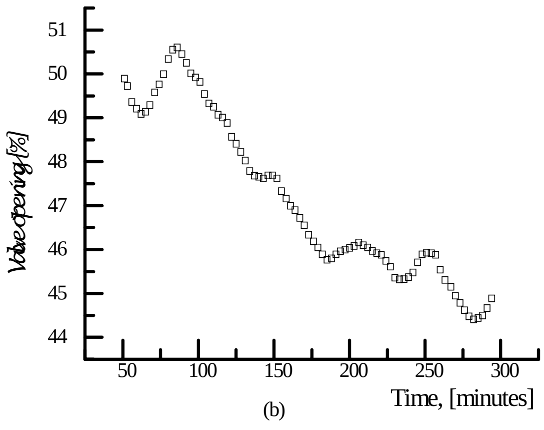


Fig. 4. Results obtained for the supervision agent of the buffer tank after training. (a) Tank hold-up evolution. (b) Valve opening.