An Approach to Tune PID Fuzzy Logic Controllers Based on Reinforcement Learning

Hacene Rezine[1], Louali Rabah[1], Jèrome Faucher[2] and Pascal Maussion[2]
[1]*Unit of Control, Robotic and Productic Laboratory, Polytechnical Military School*
[2]*Ecole Nationale d'Electrotechnique, d'Electronique, d'Informatique et d'Hydraulique de Toulouse*
Algeria

1. Introduction

In the traditional control theory, an appropriate controller is designed based on a mathematical model of the plant under the assumption that the model provides a complete and accurate characterization of the plant. However, in some practical problems, the mathematical models of plants are difficult or time-consuming to be obtained because the plants are inherently nonlinear and/or exhibit uncertainty. Thus, new methods are proposed to process these caracteristics [1]. In recent years, increased efforts have been centered on developing intelligent control systems that can perform effectively in real-time. These include the development of non-analytical methods of Artificial Intelligence (AI) such as neural networks, fuzzy logic and genetic algorithms [1] but their combinations are also introduced such as Neuro-Fuzzy and Genetic-Fuzzy techniques [2], [3]. Fuzzy logic is a mathematical approach which has the ability to express the ambiguity of human thinking and translate expert knowledge into computable numerical data. It has been shown that fuzzy logic based modeling and control could serve as a powerful methodology for dealing with imprecision and non-linearity efficiently **[**4**].** Also, for real-time applications, its relatively low computational complexity makes it a good candidate.Therefore, fuzzy logic control has emerged as one of the most successful nonlinear control techniques. Fuzzy Logic Controllers (FLC) are based on *if – then* rules integrating the valuable experiences of human. These rules use linguistic terms to describe systems. The mechanism of a FLC is that the uncertainty is represented by fuzzy sets and an action is generated co-operatively by several rules that are triggered to some degree, and produce smooth and robust control outputs. Recently, many authors proved that it is possible to reproduce the operation of any standard continuous controller using fuzzy controller [5] - [8].

Fuzzy logic controllers has shown good performances on the controlling of the complex, ill-defined and uncertain systems [9] and are being used siccessfully in many application areas such as mobile robots, subway system, nuclear reactor control and automobile transmission control, etc .

During the building of the FLC, the important tasks are the structure identification and parameters tuning [10]. The structure identification of the FLC includes the input-output variables of a controller, the rule base, the determination of the number of rules, the

Source: Robotics, Automation and Control, Book edited by: Pavla Pecherková, Miroslav Flídr and Jindřich Duník, ISBN 978-953-7619-18-3, pp. 494, October 2008, I-Tech, Vienna, Austria

antecedent and consequent membership functions and their partition on their spaces respectively, the inference mechanism and the defuzzification method. The parameters tuning includes determing the optimal parameters of membership functions antecedent and consequent but also the scaling factors. [11].
The main problem arises from there not being a systematic approach to improve system performance. In conventional approach, the problem of generation of rules is solved by exploiting the knowledge of an expert or obtaining knowledge base (i.e, training data) by investigating relationship between an existing controller and the target system and forming the rule-base by a trial-and-error approach. An important number of choices is given a priori, these choices are carried with empirical methods, and then the design of the FLC can prove to be long and delicate towards the important number of parameters to determine, and can lead then to a solution with poor performance [12].
With this subjective approach, it is difficult for a designer to examine complex systems to find the necessary number of rules, and to determine appropriate parameters of the rules for implementing the fuzzy controller [13]. Also, it isn't easy to design an optimized fuzzy controller. Therefore, there has been a strong motivation to automate this process and consequently many researchers have been working to find learning algorithms for fuzzy system design.
Several approaches have been presented to learn and tune the fuzzy rules to achieve the desired performance. These automatic methods may be divised into two categories of supervised and unsupervised learning by whether the teaching signal is needed or not.
In the supervised learning approach, at each time step, if the input-output training data can be acquired, the FLC can be tuned based on the supervised learning methods. The artificial neural network (ANN)-based FLC can automatically determines or modifies the structure of the fuzzy rules and parameters of fuzzy membership functions with unsupervised or supervised learning by representing a FLC in a connectionist way such as ANFIS or other [14]- [17].
The other category contains genetic algorithm (GA) [18]-[23] and reinforcement learning (RL) systems [24]-[26] which are unsupervised leaming algorithms with the self-learning ability [9]. The GA-based and RL-based FLCs are two equivalent learning schemes which need a scalar response from the environment to provide the action performance [28], *that* value is easier to collect than the desired-output data pairs in the real application [11].
The difference between the GA-based and RL-based FLCs lies in the manner of state-action space searching. The GA-based FLC is a population based approach that encodes the structure and/or parameterof each FLC into chromosomes to form an individual, and evolves individuals across generations with genetic operators to find the best one. The RL-based FLC uses statistical techniques and dynamic programming methods to evaluate the value of FLC actions in the states of the world. However, the pure GA-based FLC can not proceed to the next generation until the arrival of the external reinforcement signal and dit is not easy pratical in real time applications. In contrast, the RL-based FLC can be employed to deal with the delayed reinforcement signal that appeares in many situations [11]. Recently, some researches on combining the advantages of GAs and RL have been proposed [28]-[30].
The basic idea of the reinforcement learning is to learn, through trial-and-error interaction with a dynamic environnement which returns a critic, called reinforcement, which can be thought of as a reward or a punishment, the control actions to determine desired changes in the control output that will increase the index of performance. Reinforcement learning

techniques assume that, during the learning process, no supervisor is present to directly judge the quality of the selected control actions, and therefore, the final evaluation of process is only known after a long sequence of action. Also, the problem involves optimizing not only the direct reinforcement, but also the total amount of reinforcements the agent can receive in the future.This leads to the temporal credit assignment problem, i.e., how to distribute reward or punishment to each individual state-action pair to adjust the chosen action and improve its performance [31].

Supervised learning is more efficient than the reinforcement learning when the input-output training data are available [32], [33]

However, in most real-world application, precise training data is usually difficult and expensive to obtain or may not be available at all [12].

For the above reasons, reinforcement learning can be used to tune the fuzzy rules of fuzzy systems. Kaelbling, littman and Moore [34], and more recently Sutton and Barto [35], characterize two classes of methods for reinforcement learning: methods that search the space of value functions and methods that search the space of policies. The former class is exemplified by the temporal difference (TD) method and the latter by the genetic algorithm (GA) approach [36]**.** To solve reinforcemnt learning problem, the most approach is TD method [37]-[39]. Two TD based reinforcement learning approaches have been proposed the Adaptive Heuristic Critic (AHC) [40], [41] and Q-learning [42], [43]. The AHC consists of two separate networks: an action network actor) and an evaluation network (critic). Based on the AHC, many learning approaches have been proposed [20], [26], [40], [44]. One drawback of these actor-critic architectures is that they usually suffer from the local minimum problem in network learning due to the use of gradient descent learning method.

Besides the aforementioned AHC algorithm based learning architecture, more and more advances are being dedicated to learning schemes based on Q-learning [45]. Some Q-learning based reinforcement learning structures have also been proposed [46] - [52]. Q-Learning is also modified to Dyna [53], TPQ-Learning [54], CQ-Learning [55], Q(λ)-Learning [56], and so on. Glorennec and Jouffe [51],[52],[57] extented the original Q-Learning method into a fuzzy environnment and introduced two fuzzy reinforcement learning methods, i.e., Fuzzy Actor-Critic Learning (FACL) and Fuzzy Q-Learning (FQL), to select the optimal conclusion for each fuzzy from an associated discrete action set. In these methods, the antecedent parameters are set using the *a priori* task knowledge of the user. From the point of view of reinforcement learning, a fuzzy inference system (FIS) is a means to introduce generalization in the state space and generate continuous actions in the reinforcement-learning problem whereas from the point of view of FISs, reinforcement learning is a learning method used to tune a fuzzy controller in a flexible way [58]. Fuzzy Q-learning collapses the two measures used by fuzzy actor/critic algorithms into one measure referred to as the Q-value []. It may be considered as a compact version of the FACL, also we adopt Fuzzy *Q*-learning in this work because it is conceptually simpler in implementation, and has been found empirically to converge faster in many cases [59], [60], for each fuzzy rule, a *q* value is defined for each fuzzy consequence, which is the estimated cumulative reward for the fuzzy antecedents and fuzzy consequence pair of the rule. Q-learning is used to update these *q* values. Optimal or sub-optimal FLC can be constructed by choosing the fuzzy consequence with the highest *q* value for each rule. However the predefined value set needs to be set up by human experts and it is kept unchanged during learning, also if an improper value set is assigned, then those algorithms may not succeed at all [48], [50].

Horiuchi and *al* [49] consider a similar algorithm, termed fuzzy interpolation based Q-learning and further propose an extended roulette selection method so that continuous-valued actions can be selected stochastically based on the distribution of Q-values [61] proposes another version of Q-learning dealing with fuzzy constraints. In this case, we do not have fuzzy rules, but “fuzzy constraints” among the actions that can be done in a given state. These works, however, only adjust the parameters of FIS online. Structure identification, such as partitioning the input and output space and determination of the number of fuzzy rules are still carried out offline and it is time consuming. In [4] a novel online self-organizing learning algorithm is developed so that structure and parameters identification are accomplished automatically and simultaneously based only on Q-learning. In [45], [48], a dynamic fuzzy Q-learning is proposed for fuzzy inference system design. In this method, the consequent parts of fuzzy rules are randomly generated and the best rule set is selected based on its corresponding Q-value based genetic reinforcement learning. The problem in these approachs [4], [45], [50] is that if the optimal solution is not present in the randomly generated set, then the performance may be poor.
In order to solve these problems, this paper provides a systematic procedure for designing Fuzzy PID (FPID) controllers based on a reinforcement learning method. It is an automatic method capable of self-tuning parameters of a FLC based only on reinforcement signals. Continuous states are handled and continuous actions are generated by fuzzy reasoning. Prior knowledge can be embedded into the fuzzy rules, which can reduce the training time significantly. The proposed method is an efficient learning method whereby not only the conclusion part of a FLC can be tuning online, but also the parameters of antecedent part of a FLC can be tuning. We employ this approach for testing output voltage control of a DC/DC buck converter which is a traditional benchmark for testing nonlinear controllers, due to their inherent nonlinear characteristics [62], [63].
The best-known industrial process controller is the proportional-integral-derivative (PID) controller because of its simple structure, easy of design, inexpensive maintenance, low cost, and robust performance in a wide range of operations. However, it has been known that conventional PID controllers generally do not work well for nonlinear systems, higher order and time-delayed linear systems, and particularly complex and vague systems that have no precise mathematical models. To overcome these difficulties, the FPID controllers were developed and their improvement is still investigated [64]-[82]. This paper is devoted to this problem and describes some of the design aspects of the FPID.
The key concept of the proposed learning scheme is to evaluate all the principal parameters FPID in a procedure in three stages. The idea is to start with a basic FPID controller with its structure is chosen *a* priori and fixed during learning. In this work, we employ a Takagi–Sugeno of order zero as the controller of the system and the parameters tuning of fuzzy controler are the main issue researched. The membership functions or consequent parameters of each input/ouput variable are determinated with an equidistant partition. The neccessary scaling factors of the basic FPID are deduced from an initially one open-loop experimental response indicial as Ziegler-Nichols or Broida methods. This simple experimental on-site can be thought of as initial knowledge of the system and this basic FPID controller can yield an action that is feasible but far from optimal. In view of this, Reinforcement Learning is added to tune the fuzzy controller online. The predefined settings are used as starting points, also it is possible to determine the optimal parameters without too many iterations and the system can be operated safely even during learning.

Also in stage second FQL algorithm is used to select the optimal parameters of the FPID from an finite discrete set around the precedent predefined settings. This can be thought of as roughly tuning. The FQL algorithm proposed by Jouffe [52] is here extended for the antecedent parameters. Finally, in the third stage, a fine tuning procedure is follow to improve the FPID performance. This fine tuning is developped into an architecture composed of two integrated feedforward networks is proposed. One network (Q estimator **QE-FIS)** acts as a critic network to guide the learning of the other network (the action network). The action network is our FPID controller. Using the temporal difference (**TD**) prediction method, the critic network can predict the external reinforcement signal and provide a more informative internal reinforcement signal to the action network. The action network uses the gradient-descent algorithm to adapt itself in continuos according to the internal reinforcement signal. With the proposed architecture, the best parameters of the FPID in considering an IAE-criterion are determinated. This stage can be seen of as fine tuning and this way can solve the local minima problem. As a result, unlike many fuzzy Q-learning approaches that select the optimal action based on finite discrete actions [83]-[85], our proposed algorithm allows to obtain a continuos control output, the agent to learn more effectively, and helps reduce the time spent acting randomly. The salient features in our method are: (1) antecedents parameters of the fuzzy rules also can be updated, (2) not only discrete-valued antecedents/consequents parameters but also continuous-valued can be treated. (3) our technique can be used a precise simulator to speed up the learning process after final learning is achieved through the real system. Simulation and experiment results of a DC/DC Buck Converter indicate that efficiency and effectiveness of the proposed approach. Furthermore, the FPID controller learned by this approach has robust and adaptability, and can be applied to the different environments.
This paper is organized as follows. Section II briefly introduces reinforcement learning. The implementation and the limits of the Fuzzy-Q-Learning algorithm are introduced in section III. The architecture of the controller is described in section IV. The learning algorithms and parameter update laws are presented in section **V.** Section VI illustrates the performance of our proposed method through a static converter and compares experimental results with related works. Finally, conclusions and prospects are drawing in section VII.

2. Fuzzy PID system presentation

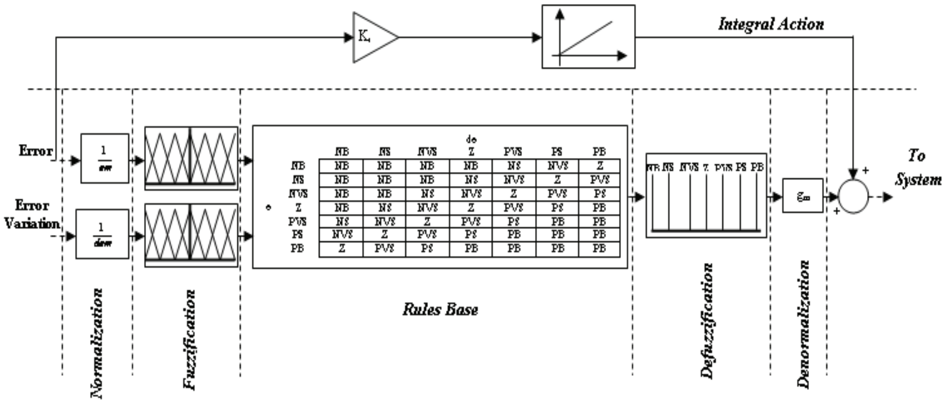
2.1 Control structure

The aim of this paper is the implementation of a FPID controller achieving the following properties:

1. Robustness around the operating point (e.g., in the case of a load change);
2. Good dynamic performance (i.e., rise time, overshoot, settling time, and limited output ripple) in the face of input voltage variations (and load changes);
3. Invariant dynamic performance in presence of varying output operating points.

We use a FPID based on the Takagi-Sugeno-type zero order method. In the FLC literature many forms of FPID structures [66], [71], [74] have been proposed. The controller in our work is a simple and classical FPID controller drawn on Fig.1. It is divided in two parts: a fuzzy part which performs the proportional-derivative action and a crisp integrator which is placed parallel of this fuzzy part so as to ensure a zero steady state error. Such a FPID

combines a high effciency with implementation easyness and is userfriendly because of its FPID-like action.
The two inputs of the controller are the error *e(k)* between the reference signal *y** and the measured signal *y* and the variation of this error *de(k)*. The output variable is the change in the control quantity $\Delta c_n(k)$. So as to ensure a good portability of the FPID controller, normalisation factors called *em*, *dem* and *gm* are used (Fig.1).



e \ de	NB	NS	NVS	Z	PVS	PS	PB
NB	NB	NB	NB	NB	NS	NVS	Z
NS	NB	NB	NB	NS	NVS	Z	PVS
NVS	NB	NB	NS	NVS	Z	PVS	PS
Z	NB	NS	NVS	Z	PVS	PS	PB
PVS	NS	NVS	Z	PVS	PS	PB	PB
PS	NVS	Z	PVS	PS	PB	PB	PB
PB	Z	PVS	PS	PB	PB	PB	PB

Fig 1. Structure of the Fuzzy PID Controller

The FPID considered here uses triangular membership functions with strong fuzzy partition because of its facilties and excellent approximation properties, and have been shown to be sufficient in a number of applications. We adopt for the two normalised input values $e_n(k)$ and $de_n(k)$ seven triangular membership functions on each input and seven singletons at the output. We use the Mac Vicar-Whelan 'base rules (1977) with 49 fuzzy rules. The membership functions are called PB, PS, PVS, Z, NVS, NS andNB (P: Positive, N: Negative, B: Big, S: Small, VS: Very Small). So as to warrant a similar response of the system for positive or negative sollicitations, a zero-symmetry can be imposed for both input membership functions and output singletons and a classical antidiagonal rules table is used. In addition, for a good portability of the FPID, the two inputs and the output are normalized on a [-1, +1] universe of discourse. Furthermore, the FPID is supposed to be well-normalized, it implies that the position of PB's and NB's apex are assigned to ±1 respectively. For the two inputs and the output of the FPID, the positions of the PS's and PVS's membership function's apex are mobile. Furthermore, the and-method is based on the product and the defuzzyfication method on the center of gravity is considered.

2.2 Factors definitions

In this part, we suppose that we don't know the dynamic of the system to control. Even for such a very simple fuzzy control structure, the tuning parameters of such a FPID are very numerous (positions of membership functions, normalisation gains, fuzzy rules, …). In this paper and so as to limit the number of tuning parameters, we just hold 15 or 26 of these parameters back, which contributions to the optimization process according to the IAE criterion seam to be the greatests. These 15 or 26 parameters, that constitute the set of controlable factors, are the following:

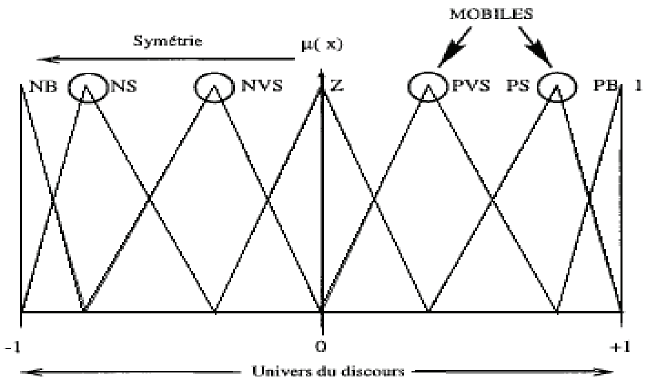


Fig 2. Membership functions for the two inputs e, de

- On the input *e* of the FPID: the position of the membership functions apex PS and PVS, the position of the membership functions apex NS and NVS is obtained by symmetry (fig.2.). The normalization factor em is supposed to be equal to the magnitude of the step sollicitation.
- On the input *de* of the FPID: the position of the membership functions apex PS and PVS and the position of the membership functions apex NS and NVS is obtained by symmetry.
- On the output Δc_n of the FPID: the positions of the PS's and PVS's singletons. The principle of reinforcement learning allows considering for each fuzzy rule an individual discrete action set. Also at the end of the learning processus, the same linguistic label (Table I.4.) can have other significance in the base rule. Consequently, we obtain 11 or 22 tuning parameters in the case of a classical antidiagonal rules table or not respectively.

The normalization factor de_m, the denormalization gain g_m and the integrator gain K_i are fixed during the learning processus. They are determited by an open-loop identification test.

3. Q-learning algorithms

As previously mentioned, there are two ways to learn either you are told what to do in different situations or you get reward or punishment for doing good respectively bad actions. The former is called supervised learning and the latter is called learning with a critic, of which reinforcement learning (RL) is the most prominent representative with the self learning ability. It is shown that supervised learning is more efficient than reinforcement learning [32]. However, reinforcement learning only needs the critic information (evaluative signal) with respect to the different states of the controlled system [35]. This evaluative signal contains much less information than the reference signal used in supervised learning; also the reinforcement learning is appropriate for systems operating in a knowledge-poor environment [28].

The basic idea of reinforcement learning is that agents learn behaviour through trial-and-error interactions with the controlled system, and receive a critic, called reinforcement, which can be thought of as a reward or a punishment for behaving in such a way that a goal is fulfilled. This learning method is based on the common-sense idea that if an action is followed by a satisfactory state, or by an improvement, then the tendency to produce that action is strengthened, i.e., reinforced [58]. Reinforcement learning does not need teacher signal to guide action, since the learner is not told which action to take, it must discover the policy most effective, i.e. to know, in each possible situation, which action is achieved to maximize the expected cumulative reward in the long-term. In reinforcement learning, the final evaluation of process can be only known after a long sequence of actions. Thus, an internal evaluation function that is more informative than the evaluation function by the external critic is considered. This internal evaluation function takes the form of the expected sum of infinite horizon discounted payoffs, called the evaluation value of a policy:

$$R = \sum_{t=0}^{\infty} \gamma^{k} r_{t} \qquad (1)$$

Where γ is the discount factor (*0* ≤γ≤*1*) used to determine the present value of future rewards.and r_t is the external reinforcement signal received at time *t*.
The idea of Reinforcement Learning can be generalized into a model, in which there are two components: an agent that makes decisions and an environment in which the agent acts. For every time step *t*, the agent is in a state $s_t \in S$ where *S* is the set of all possible states, and in that state the agent can take an action $u_t \in (U_t)$, where (U_t) is the set of all possible actions in the state s_t. As the agent transits to a new state s_{t+1} at time *t* + 1 it receives a numerical reward r_{t+1},. It up to date then its estimate of the evaluation function of the action $Q(s_t, u_t)$ using the immediate reinforcement, r_{t+1}, and the estimated value of the following state, $Q_t * (s_{t+1}, u_t^{'})$, which is defined by:

$$Q_t * \left(s_{t+1}, u_t^{'}\right) = \max_{u'_t \in U_{s_{t+1}}} Q\left(s_{t+1}, u^{'}\right) \qquad (2)$$

The Q-value of each state/action pair is updated by

$$Q_{t+1}(s_t,u_t) = Q_t(s_t,u_t) + \beta\left\{r_{t+1} + \gamma Q_t * \left(s_{t+1},u_t^{'}\right) - Q_t(s_t,u_t)\right\} \qquad (3)$$

Where $r_{t+1} + \gamma Q*_t(s_{t+1}, u_t') - Q_t(s_t, u_t)$ is the temporal difference (TD) error and β is the learning rate. This algorithm is called Q-Learning. It shows several interesting characteristics. The estimates of the function Q, also called the Q-values, are independent of the policy pursued by the agent. To calculate the evaluation function of a state, it is not necessary to test all the possible actions in this state but only to take the maximum Q-value in the new state (eq.4). However, the too fast choice of the action having the greatest Q-value:

$$u'_t = \arg \max_{u_t \in U_{s_{t+1}}} Q_t\left(s_{t+1}, u_t\right) \qquad (4)$$

Can lead to local minima. To obtain a useful estimate of Q, it is necessary to sweep and evaluate the whole of the possible actions for all the states: it is what one calls the phase of exploration [35].

In the previous equation, at state, only the Q-value of the activating action is updated, and the actions taken during the past time steps are not considered. But as is often the case with real, a reinforcement signal may not be available at a time long after the occurrence of a sequence of actions. This requires improving long-term consequences of an action or of a strategy for performing actions, in addition to short-term consequences. This problem is known as temporal credit assignment problem, i.e., how to distribute reward or punishment to each individual state/action pair to adjust the chosen action and improve its performance Also, to speed learning Sutton [35] extended the evaluation in all the states, according to their eligibility traces that memorise the previously visited state/action pairs weighted by their proximity to time step. They work like a short-term memory process activated by the occurrence of state/action pairs. The eligibility traces combined with Q-learning can be defined in several ways [48], [51]. Accumulating eligibility is defined by:

$$e_t(s)=\begin{cases}1+\gamma\lambda e_{t-1}(s) & if \quad s=s_t\\ \gamma\lambda e_{t-1}(s) & \text{otherwise}\end{cases} \tag{5}$$

where λ is the eligibility rate used to weight time, since it accumulates whenever a state/action pair is selected and decays gradually when is not selected.
The algorithm Q (λ) is a generalization of Q-Learning (when λ=0) which uses the eligibilities traces. With eligibility trace, equation (3) is changed to:

$$Q_{t+1}(s_t,u_t)=Q_t(s_t,u_t)+\beta\left\{r_{t+1}+\gamma Q^*_t(s_{t+1},u_t')-Q_t(s_t,u_t)\right\}e_t(s) \tag{6}$$

4. Fuzzy Q-learning algorithms

The discrete Q-Learning such as we described it uses a discrete space of states and actions which must be have reasonable size to enable the algorithms to converge in an acceptable time in practice. In this case, a look-up table can be built up by listing the state/action pairs with their *Q* values. However in many applications, the number of state/action pairs is very large. Thus a method that is able to make Q-Learning applicable to the continuous problem domain is necessary. The Fuzzy Inference System (FIS) learner is one existing generalization methods which can be introduce generalization in the state space and generate continuous actions in the reinforcement-learning problem.

A- FQL algorithm for consequents part (discrete parametrs tuning)

The principle of Fuzzy Q-Learning (FQL) proposed by Jouffe [51] is reinforcement learning method that tunes with only the consequent part in an incremental way based only on reinforcement signals. Each fuzzy rule *Ri* has possible *k* discrete candidate consequents (actions) $U_i=\left(u_1^i,\ u_2^i,.....,\ u_k^i\right)$ and it memorizes the parameter vector q associated with each of these actions. Local actions $(u_1,.......,u_k)$ selected from *U* compete with each other based on their q-values so as to maximize the discounted sum of rewards obtained while achieving the task. Each *Rule R_i* of the FPID can be described as follow:

If e is L_1^i and *de* is L_2^i then u^i is u_1^i with $q(S, u_1^i)$
or u^i is u_2^i with $q(S, u_2^i)$ L_m^i is linguistic label related to each
- - - - - - - - - -
or u^i is u_k^i with $q(S, u_k^i)$

input state, u_j^i is a rule consequent (action) which corresponds to the consequent part of i-th fuzzy rule R_i and has its own q-value q_j^i.

1-*Generation of Continuous Actions*

The current state S_t is perceived by the means of its activated degree of its firing rules. The winning local action cooperates to produce the global continuos output action which is defined by the weighted sum of the local actions elected in the fired rules that describe this state. It is given by

$$U_t(S_t) = \sum_{i=1}^{n} \pi_U(S_{t,}q_t)\alpha_t^i = \sum_{i=1}^{n} u_t^i \alpha_t^i \tag{7}$$

Where u_t^i is the selected action of rule $R_{i,}$ at time step t by a policy $\pi_U(S_{t,}q_t)$ and α_t^i is the rule's normalized firing strength.

After application of the global action, $U_t(S_t)$, the states change to S_{t+1}.

Optimal or sub-optimal FPID can be constructed by choosing the fuzzy consequence with the highest q value for each rule. But, at the beginning stage for training, q-value can not correctly describe the valuation of action and taking the greedy actions or exploiting the previous experience too much during the learning would lead to local optima. The learnin algorithm would fail to find good actions On the other hand, taking random actions or exploring the spaces too much would affect both the learning convergence and the learning rate. Therefore in order to explore the set of possible actions and acquire experiences through the reinforcement signals, the actions are selected using an Exploration-Exploitation strategy. There are some random policies and the Boltamann probability distribution and -greedy method are the effective exploration/exploitation policy (EEP) to choose action. The algorithm FQL proposed in this paper uses an EEP [50], [51], combining an undirected exploration part $\rho^i(u)$ and directed exploration part $\eta^i(u)$ which are introduced by a random vector and a counter associated to actions. The proposed exploration-exploitation policy selects a local action from possible discrete actions vector, as follows:

$$\pi_U(S_t, q_t) = \arg\max_{u \in U}(q_t(S_t, u) + \eta(S_t, u) + \rho(S_t, u)) \tag{8}$$

The undirected term of exploration η stems from a vector of random values Ψ, (exponential distribution) scaled up or down to take into account the range of q values.

$$s_f = \begin{cases} 1 \quad if \quad \max_u(q\ (S_t,u)) = \min_u(q(S_t,u)) \\ \dfrac{s_p(\max_u(q(S_t,u)) - \min_u(q(S_t,u\ )))}{\max(\psi)}, \quad otherwise, \end{cases}$$

$$\eta(S_t, u) = s_f \psi \tag{9}$$

where s_p is the noise size, with respect to the range of q qualities, and s_f is the corresponding scaling factor. Decreasing the factor s_p implies reducing the undirected exploration
The directed term ρ gives a bonus to the actions that have been rarely elected

$$\rho(S_{t,} u) = \frac{\theta}{e^{n_t(S_t, u)}} \tag{10}$$

where θ represents a positive factor used to weight the directed exploration and $n_t(S_t,u)$ is the number of time steps in which action u has been elected. This term is not directly available; it is approximated by the following fuzzy inference:

$$n_t(S_t, u) = \sum_{i=1}^{n} n_t(u^i)\alpha_t^i \tag{11}$$

where $n_t(u^i)$ is the number of applications, at time step t, of action u in rule R_i. Let be u_t^i the elected discrete action at time step t in rule R_i. $n_t(u^i)$ is then defined according to

$$n_t(u^i) = n_{t-1}(u^i) + 1, \tag{12}$$

2-.*Update of Q-Values*
We define also a function Q, which gives the action quality with respect to states. Q-values are also obtained by the FPID outputs, which are inferred from the quality of local discrete actions that constitute the global action.

$$Q_t(S_t, U_t) = \sum_{i=1}^{n} q_t(S_t, u_t^i)\alpha_t^i \tag{13}$$

where U_t is the global continuos action, u_t^i is the selected action of rule R$_i$ at time step t and q_t is the q-value associated with fuzzy state, S$_i$ and action, u_t^i

Based on TD Learning, the Q-values corresponding to the rule optimal action are used to estimate the TD error, which is defined as follows :

$$Q^*_t\left(S_{t+1}\right) = \sum_{i=1}^{n}(\max_{u' \in U} q_t\left(S_{t+1}, u'_t\right)\alpha_t^i \tag{14}$$

and the TD error is defined only with quantities available at time step $t + 1$ as follow

$$\tilde{\varepsilon}_{t+1} = r_{t+1} + \gamma Q^*_t(S_{t+1}) - Q_t(S_t, u_t) \tag{15}$$

This TDerror can be used to evaluate the action just selected. If the TD error is positive, it suggests that the quality of this action should be strengthened for future use, whereas if the TD error is negative, it suggests that the quality should weakened [4]. The learning rule by taking the eligibity traces is given by

$$q_{t+1}(S_i,u_t^i) = q_t(S_i,u_t^i) + \beta_t^i \tilde{\varepsilon}_{t+1} \alpha_t^i e_t(S_i,u_j) \tag{16}$$

Where β_t^i is the adaptative learning rate of rule R_i at time step t which is updating as follow:

$\delta_t^i = \tilde{\varepsilon}_{t+1}.\alpha_t^i$ Integrates eligibility trace

$$\beta_{t+1}^i = \begin{cases} \beta_t^i + k & if \quad \bar{\delta}_{t-1}^i.\delta_t^i > 0, \\ \beta_t^i(1-\psi) & if \quad \bar{\delta}_{t-1}^i.\delta_t^i < 0, \\ \beta_t^i & otherwise \end{cases} \tag{17}$$

$\bar{\delta}_t^i = (1-\varphi)\delta_t^i + \varphi\bar{\delta}_{t-1}^i$ Represents the geometric average

β_t^i is used in order to increase the convergence rate during the FPID approximation and prevent oscillations and it is based on the Delta-Bar-Delta learning rule [51]. The rule (17) increments the critic learning rates linearly to prevent them from becoming too large too fast and decrements them exponentially to ensure that they always stay positive and to enable them to decrease rapidly

and $e_t(S_i,u_j)$ is the trace associated with discrete action u_i of rule R_i at time step t,

$$e_t(S_i,u_j) = \begin{cases} \gamma\lambda e_{t-1}(S_i,u_j) + \alpha_t^i & if \; u_i = u_t^i \\ \gamma\lambda e_{t-1}(S_i,u_j), & \text{otherwise} \end{cases} \tag{18}$$

The traces are updated between the action computation and its application.

B- FQL algorithm for antecedents part (discrete parametrs tuning)

In the previous method, the antecedent parameters are set using the *a priori* task knowledge of the user. To restrict a FPID optimization to the only tuning of the parameters of the consequent part is often insufficient to reach high performances. Also in this paper, we bring a slight difference to the algorithm proceeding by introducing an extension to the Fuzzy Q-Learning algorithm to also allow the online tuning of the parameters of the antecedent part (positions of input fuzzy sets).

The principle of the FQL algorithm applied to the antecedent parameters will consist in selecting for each membership function a modal point from a possible discrete candidate modal points set basing on the evaluation function of the action $Q(s_t,u_t)$ because as the equations (7, 13) show it, the calculation of the action U and its quality Q(S,u) is closely linked to the antecedent parameters by the means of the degrees of activation of the fuzzy rules α_t^i. The algorithm principle is depicted in Fig.3, 4.

Let us consider the FPID controller with the two input variables *e* or *de*. Each universe of discourse of *e* or *de* is partitioned in *Nmf*$_i$ membership functions

$F_i^j :\{F_i^1, F_i^2, \cdots, F_i^j, \cdots, F_i^{Nmfi}\}$ with each membership function. F_i^j has an possible discrete modal points set $A(i,j) : \{a_i^{j,1}, a_i^{j,2} \cdots, a_i^{j,k}, \cdots, a_i^{j,Nniv_{i,j}}\}$, where $a_i^{j,k}$ is the k^{eme} point modal possible of the j^{eme} membership function of the input i F_i^j and $Nniv_{i,j}$ is the cardinal of the set. Each *Rule* R_i of the FPID can be described as follow:

$$\begin{array}{llll} \textit{If } e \textit{ is } F_1^i(a_1^{i,1}) & \textit{and } de \textit{ is } F_2^i(a_2^{i,1}) & \textit{Then } u \textit{ is } & u_t^i \\ \quad \textit{or } F_1^i(a_1^{i,2}) & \quad \textit{or } F_2^i(a_2^{i,2}) & & \\ \quad \cdots\cdots\cdots & \quad \cdots\cdots\cdots & & \\ \quad \textit{or } F_1^i(a_1^{i,k}) & \quad \textit{or } F_2^i(a_2^{i,k}) & & \end{array}$$

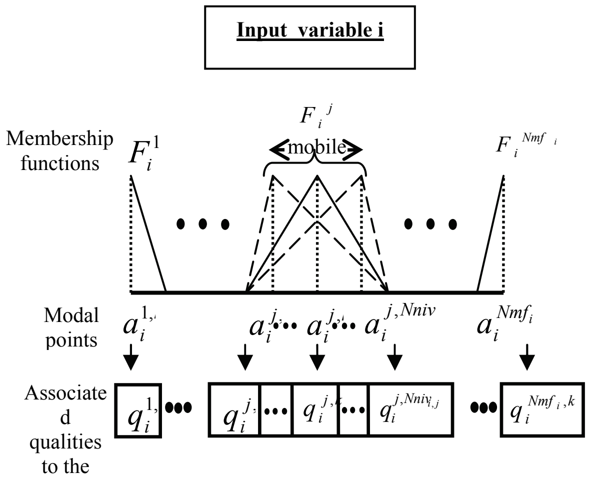


Fig. 3. FQL Structure for the antecedent part

As for the previous algorithme (section IV-A) local modal points ($a_1, \ldots\ldots, a_k$) selected from *A* compete with each other based on their q-values so as to maximize the discounted sum of rewards obtained while achieving the task. Global quality *Q* for the state S_t is then defined by the inference of these qualities locally elected

$$Q_t(S_t) = \sum_{i=1}^{2} \sum_{j=1}^{Nmf_i} q_t(a_i^j) \cdot \mu_t^{F_i^j}(x_i) \tag{19}$$

where a_i^j is the elected modal point at the time step t

for the membership function F_i^j by an exploration-exploitation policy and $\mu^{F_i^j}(x_i)$ is the membership degree of the input variable x_i ($x_1=e$, $x_2=de$) to F_i^j and it measures the contribution of the modal points to the generation of the globale action .

Execution Procedure

In order to describe the working principle of the FQL algorithm applied to the antecedent parameters, a one-time-step global execution procedure is presented. Let *t+1* be the current time step; the learner has performed the action U_t and has received a reinforcement signal r_{t+1}according to the state transition S_t to S_{t+1}. After the fuzzification step and rule strength computations (describing the new state) by using the modal points (a_t) elected with the step of previous time *t*,, the eight stages are as follows:

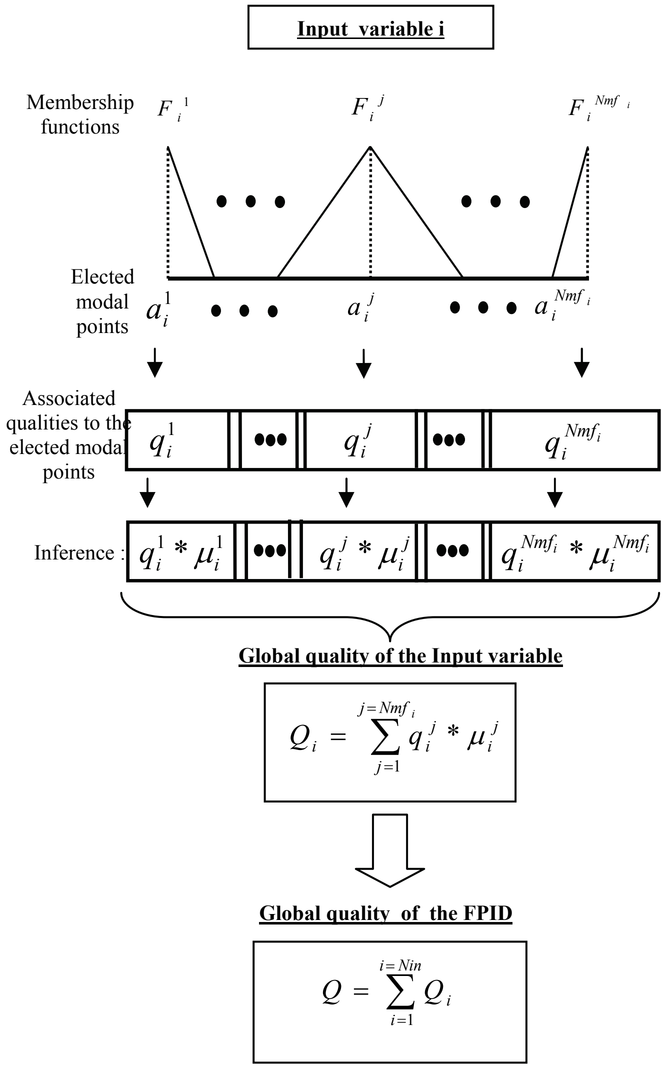

Input variable i
Membership functions
F_i^1
F_i^j
$F_i^{Nmf_i}$
Elected modal points
a_i^1
a_i^j
$a_i^{Nmf_i}$
Associated qualities to the elected modal points
q_i^1
q_i^j
$q_i^{Nmf_i}$
Inference :
$q_i^1 * \mu_i^1$
$q_i^j * \mu_i^j$
$q_i^{Nmf_i} * \mu_i^{Nmf_i}$
Global quality of the Input variable
$Q_i = \sum_{j=1}^{j=Nmf_i} q_i^j * \mu_i^j$
Global quality of the FPID
$Q = \sum_{i=1}^{i=Nin} Q_i$

Fig. 4. FQL Algorithm Principle for the antecedent part

1. Estimation of the t-optimal evaluation function corresponding to the current state from the best modal points of each membership function, as follows. :

$$Q_t^*(S_{t+1}) = \sum_{i=1}^{2} \sum_{j=1}^{Nmf_{i,j}} \left(\underset{k=1,\mathrm{Nniv}_{i,j}}{Max} \left(q_t \left(a_i^{j,k} \right) \right) \right) \cdot \mu_t^{F_i^j}(x_i) \tag{20}$$

2. Compute the TD error ;

$$\tilde{\varepsilon}_{t+1} = r_{t+1} + \gamma Q_t^*(S_{t+1}) - Q_t(S_t) \tag{21}$$

3. Update the adaptative learning rate $\vartheta_i^{j,k}(t+1)$ with the Delta-Bar rule by using the equation defined in (17) ;
4. Tune parametr vector q based on on current eligibility trace ;

$$q_i^{j,k}(t+1) = q_i^{j,k}(t) + \vartheta_i^{j,k}(t+1) \cdot \tilde{\varepsilon}_{t+1} \cdot e_i^{j,k}(t) \tag{22}$$

where: $i = 1,\ldots,n \quad , \quad j = 1,\ldots,Nmf_i \quad , \quad k = 1,\ldots,Nniv_{i,j}$

5. Elect the modal points a_i^j for each membership function F_i^j by using an exploration-exploitation strategy identical to that used in algorithm FQL for the consequents part and defined as (8) ;
6. Estimation of the new evaluation functions for the current state *t+1* with the new modal points vector. We recompute the membership degrees of the inputs to the new membership functions ($\mu_{t+1}^{F_i^j}$).
7. Update the eligibility trace which will be used in parameter updating at the next time step:

$$e_i^{j,k}(t+1) = \begin{cases} \gamma\lambda e_i^{j,k}(t) + \mu_{t+1}^{F_i^j}(x_i), & if \quad a_i^{j,k} = a_i^j \\ \gamma\lambda e_i^{j,k}(t), & otherwise \end{cases} \tag{23}$$

Eligibility trace values need to be reset to zeros at the end of each episode ;

8. New estimation of the evaluation function corresponding to the elected modal points at the current state for the current state with the new updating parametrs:

$$Q_{t+1}(S_{t+1}) = \sum_{i=1}^{2} \sum_{j=1}^{Nmf_i} q_{t+1}(a_i^j) \cdot \mu_{t+1}^{F_i^j}(x_i) \tag{24}$$

$Q_{t+1}(S_{t+1})$ will be used in the error computation at the next time step.

5. Continuos fuzzy Q-learning algorithm (continuous parametrs tuning)

Until now, the definition of controlable factors for the FPID in the reinforcemnt learning algorithms has just been led on discrete levels [83]- [85] which are kept unchanged during learning. This is the basic idea of most existing algorithms which adopt Q-learning. In general, however, this is not case and then the algorithm may fail. Indeed, for complex systems like static converter, *priori* knowledge are not available, then it becomes difficult to determine a set of parameters in which figure the optimal controlable factors for each fuzzy rule and thus the FPID controller can't accomplish the given task through Fuzzy Q-Learning.

To ensure a fine optimization of these parameters in the vicinity of those obtained in the last section, a continue reinforcement learning algorithm [58] has been proposed. Compared to the idea in this article, we employ an FIS, in the place of a neural network, to estimated Q(S, u) which is only used to tune the parameters of the FPID. As a result, our proposed algorithm is more applicable to real systems because we combine reinforcement learning with an Fuzzy Logic Controller (FLC) and it learns more effectively, and helps reduce the time spent acting randomly.
The overall architecture of this FLC, inspired of [58] is shown in Fig. 5. It is similar to that of the FACL [51]. The proposed controller is constructed of two parts: a critic, a Q(S, u) estimator FIS (QE_FIS) and an actor, the FPID which have two main responsabilities respectively : the critic is to estimate the optimal-action value function $Q^*(S,u)$, and the actor is to get the control output based on the estimated-action value function $Q(S,u)$. There is an indirect coupling between the two. Indeed, when action U acts in the system, the state S_t transites towards the new state S_{t+1}. According to this transition, the system emits a reinforcement r_{t+1}which is used in the error TD $\tilde{\varepsilon}_{t+1}$. According to the latter, we adjust the parameters of the evaluation function $Q(S,u)$ which is then used to tuning in a continuous way the controlable parameters of the FPID in such manner to maximize quality $Q(S,u)$by a rise of gradient. The architecture of this FLC is explained in detail in the following parts of this section.

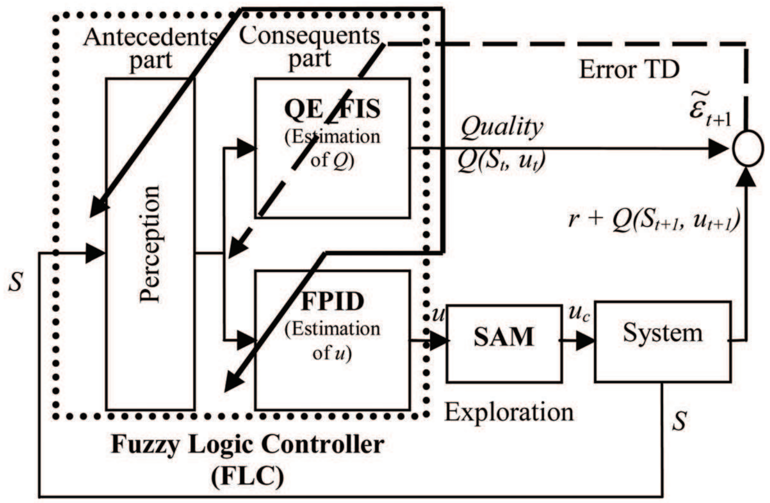


Fig. 5. Architecture proposed for the continuous tuning of the parameters FPID

A. FIS System for Estimating the Optimal-Action Value Function

As the approximator of the value function for reinforcement learning, different types of approximators have been proposed for different tasks and different objectives. Interested readers can see references indicated in [58] for more information about approximator methods. In this paper, we use a Fuzzy Inference System (FIS) to approximate and to predict the optimal-action value function $Q^*(S,u)$ due its good approximation property. The $Q^*(S,u)$estimator fuzzy (QE-FIS) is associated with the same inputs S of the FPID also they can be to amalgamate as an only one fuzzy controller which allows a considerable profit in the computing time.The QE-FIS is based on the prediction TD error $\tilde{\varepsilon}_{t+1}$ (Fig. 5) for the optimization of its parameters. It is used to guide the FPID controller to tune parameters so that the fuzzy controller will achieve better performance.

B. FPID for producing the control output

The FPID generates the control output *u* that maximizes the action value function *Q* (*S,u*) produced by the QE-FIS incrementally. The final output provided by FPID maximizes *Q* (*S,u*) with respect to all possible *u*, instead of the finite candidate discrete action set *U* in the last section. By its robustness, the FPID can achieve acceptable performance even in the early stage of learning, in which the approximation error of *Q* (*S,u*)is large [58] and the information, obtained in the section preceding, about the candidate discrete factors set *U* and *A* allow it to learn more effectively by narrowing the research space and to speed up the learning processus.

Consider a multiple-input single-output (MISO) Fuzzy Logic Controller (FLC) that performs a mapping from a state vector $\underline{x} = (x_1, x_2, \cdots, x_n)^T \in \Re^n$ to a control input $u \in \Re$. Using the Takagi–Sugeno model, the If–Then rules of the FLC may be expressed as

$$R^l : If \quad x_1 \quad is \quad F_1^l, \cdots and \quad x_n \quad is \quad F_n^l \quad Then \begin{cases} u^l = K_0^l + K_1^l x_1 + \cdots + K_n^l x_n & \rightarrow FPID \\ q = q^l \quad \rightarrow QE\_FIS \end{cases}$$

where F_i^l is the label of the fuzzy set in x_i for l= 1, …*M* (*M* is rules number), $K_0^l, K_1^l, \cdots and \quad K_n^l$ are the constant coefficient of the consequent part of the fuzzy rule and q^l is local action quality u^l. We consider in a way general the first input x_1 has Nmf_1 membership functions and the second input x_2 has Nmf_2 membership functions. In our application $n = 2, (x_1 = e \quad and\ x_2 = de)$, Nmf_1=7, Nmf_2=7, $K_0^l \neq 0, K_1^l = 0, \cdots and \quad K_n^l = 0$ have different consequents and the local quality q^l entirely qualifies the R^l rule, from where the idea to use it to tune the controlable parameters of the FPID. We use product inference for the fuzzy implication and *t* norm, singleton fuzzifier, and center-average defuzzifier; consequently, the final output value of the FPID and the final quality value of the QE_FIS are respectively

$$u(x) = \frac{\sum_{k=1}^{Nmf_1} \sum_{t=1}^{Nmf_2} \left\{ \alpha^{k,t} \cdot \left(\sum_{i=0}^{n} K_i^{k,t} x_i \right) \right\}}{\sum_{k=1}^{Nmf_1} \sum_{t=1}^{Nmf_2} \alpha^{k,t}} \tag{25}$$

$$Q(x,u) = \frac{\sum_{k=1}^{Nmf_1} \sum_{t=1}^{Nmf_2} \alpha^{k,t} \cdot q^{k,t}}{\sum_{k=1}^{Nmf_1} \sum_{t=1}^{Nmf_2} \alpha^{k,t}} \tag{26}$$

where α^l is the firing strength of the fuzzy rule R^l calculated as follow $\alpha^l = \prod_{i=1}^{n} \mu^{F_i^l}(x_i)$, μF_i^l is the membership degree of the fuzzy set F_i^l, $\left(\sum_{i=0}^{n} K_i^l x_i \right)$ is the conclusion part of the fuzzy rule R_l, ($x_0 = 1$) and q^l is the local action quality of the fuzzy rule R_l. With the choice of strong fuzzy partitions $\sum_{k=1}^{Nmf_1} \sum_{t=1}^{Nmf_2} \alpha^{k,t} = 1$.

C. Stochastic Action Modifier (SAM)

In order to guarantee that Q(x, a) converges to Q*(x, a) with a probability equal to 1, we implement an exploration strategy for the control output *u* provided by the FPID which deals with continuos actions. For that, we add a stochastic action modifier (SAM) after the FPID and before the system input [58]. The SAM generates the control command u_c, which is a gaussian random variable with mean *u* recommended by the FPID and standard deviation σ_u, and σ_u satisfies the condition that it will converge to zero gradually, *i.e.*

$$u_c = u + \sigma_u(t) \cdot n(t), and \qquad \lim_{t \to \infty} \sigma_u(t) = 0 \tag{27}$$

where *n(t)* represents a gaussian random variable normalised and centred.
The adopted standard deviation is as follow:

$$\sigma_u(t) = \exp\left(-\alpha t\right), avec \qquad \alpha \succ 0. \tag{28}$$

D. Learning algorithms for FLC

In this section, we develop the learning algorithms for the QE-FIS and the FPID. The learning mechanism (estimation of local qualities q^l) of the QE-FIS is the combination of the TD methods and the gradient descent algorithm. The learning mechanism of the FPID is based on gradient rise algorithm. Indeed, the global quality *Q* (*x, u*) (see equation 26) qualifies the action *u* but also the FPID [45]. Thus more its value is high, plus the performances of FPID are better. We adjust consequently the antecedents parameters (modal points of the membership functions) and the consequents parameters (the K_i^l coefficients parameters) of the FPID in such manner to maximize global quality *Q* (*x, u*).
While starting of a SIF optimized in the phase of discrete reinforcement learning, the proposed algorithm uses two stages to avoid all risk of instability. These two stages constitute the continuous reinforcement learning algorithm: - the first stage is the simultaneous optimization of the antecedent and consequent parameters of the QE_FIS, while fixing the consequent parameters of FPID ([values found in the phase of the discrete reinforcement learning) - the second stage is the simultaneous optimization of the parameters of the FPID and the QE_FIS, while fixing the antecedent parameters to the values found in the stage one.

1. ***Learning algorithm for consequent (QE-FIS)***

From the previous section on reinforcement learning, we know TD methods learn their estimates, in part, on the basis of other estimates. We can also tune the consequent

parameters of our proposed QE-FIS based on this method. We can achieve the task of approximating the optimal-action value function with the QE-FIS by reducing the TD error $\tilde{\varepsilon}_{t+1}$, continuously with a descent gradient algorithm.
The quality-update rule of the QE_FIS is then given by:

$$q_{t+1}^{l} = q_{t}^{l} + \eta_{t+1}\tilde{\varepsilon}_{t+1} \cdot e_{t}^{l} \tag{29}$$

The adaptative learning rate η_{t+1} and the eligibility traces for the QE-FIS are determided on the rules by using respectively the equations (17) and (18) ;
The TD error is calculated as follows:

$$\tilde{\varepsilon}_{t+1} = r_{t+1} + \gamma \max_{u'} Q(x_{t+1}, u') - Q(x_{t}, u_{t}) \tag{30}$$

In this expression, we consider that $\max_{u'} Q(x_{t+1}, u') \approx Q(x_{t+1}, u_{t+1})$ because u_{t+1} is the ouput FPID tuned continuously to maximize $Q\ (x, u)$, therefore $\tilde{\varepsilon}_{t+1}$ is written as follows:

$$\tilde{\varepsilon}_{t+1} = r_{t+1} + \gamma Q(x_{t+1}, u_{t+1}) - Q(x_{t}, u_{t}) \tag{31}$$

2. ***Learning algorithm for FPID and antecedent parameters QE-FIS***
Now, we consider how to improve the control policy using the associated value function. In other words, we consider how to tune the parameters of the FPID controller based on the approximated $Q\ (x, u)$obtained from the previous section.
In order to optimize the output of the FPID, we can update the parameters of the FPID to maximize the action value function $Q(x, u)$ with respect to the control output u for the current state. As a result, the learning algorithms of the FPID can be derived using gradient rules
2.1 ***Learning algorithm for antecedent parameters***
The parameters of the modal point vector ($a_i^j \quad i = 1,....,n \quad ; \quad j = 1,...,Nmf_i$) are updated by a rise gradient algorithm :

$$a_i^j(t+1) = a_i^j(t) + \rho \frac{\partial Q(x_{t+1}, u_{t+1})}{\partial a_i^j} \tag{32}$$

Let us consider the case of a modal point of the first input. The calculation of the $\dfrac{\partial Q(x,u)}{\partial a_1^j}$ revealed the three expressions $\mu^{F_1^{j-1}}(x_1), \mu^{F_1^j}(x_1) \ et \ \mu^{F_1^{j+1}}(x_1)$ where the modal point a_1^j can intervene in two expressions in maximum among these three. The calculation of the membership degrees of different fuzzy set $F_i^{j-1}, \ F_i^j \ et \ F_i^{j+1}$) shows that it exists four possible cases :
The table 1 recapitulates every case to be considerate.

	$\frac{\partial Q(x,u)}{\partial a_i^j} =$	Explanation of the notations	
		First input e *(i=1)*	*Second input de* *(i=2)*
1.3 case : *If* $x_i \le a_i^{j-1}$ *or* $x_i \ge a_i^{j+1}$ *or* $x_i = a_i^j$	0	$q_1^{j-1} = \sum_{l=1}^{Nmf_2} \mu^{F_2^l}(x_2)\cdot q^{j-1,l}$ $q_1^{j} = \sum_{l=1}^{Nmf_2} \mu^{F_2^l}(x_2)\cdot q^{j,l}$ $q_1^{j+1} = \sum_{l=1}^{Nmf_2} \mu^{F_2^l}(x_2)\cdot q^{j+1,l}$	$q_2^{j-1} = \sum_{k=1}^{Nmf_1} \mu^{F_1^k}(x_1)\cdot q^{k,j-1}$ $q_2^{j} = \sum_{k=1}^{Nmf_1} \mu^{F_1^k}(x_1)\cdot q^{k,j}$ $q_2^{j+1} = \sum_{k=1}^{Nmf_1} \mu^{F_1^k}(x_1)\cdot q^{k,j+1}$
2. case : *If* $a_i^{j-1} < x_i < a_i^j$	$\frac{(q_i^j - q_i^{j-1})\cdot(a_i^{j-1} - x_i)}{(a_i^j - a_i^{j-1})^2}$		
4. case : *If* $a_i^j < x_i < a_i^{j+1}$	$\frac{(q_i^j - q_i^{j+1})\cdot(a_i^{j+1} - x_i)}{(a_i^j - a_i^{j+1})^2}$		

TABLE I Summary of the calculation of $\frac{\partial Q(x,u)}{\partial a_i^j}$ Some remarks are to be noted.

1. There is a discontinuity to the point a_i^j (superior limit of F_i^{j-1} , modal point of F_i^j and inferior limit of F_i^{j+1}), indeed :

$$\lim_{x_i \to a_i^{j-}} \frac{\partial Q(x,u)}{\partial a_i^j} \neq \lim_{x_i \to a_i^{j+}} \frac{\partial Q(x,u)}{\partial a_i^j} \neq \left.\frac{\partial Q(x,u)}{\partial a_i^j}\right|_{x=a_i^j}$$

Superior limit of F_i^{j-1} : $\lim_{x_i \to a_i^{j-}} \frac{\partial Q(x,u)}{\partial a_i^j} = \lim_{x_i \to a_i^{j-}} \left\{ \frac{(q_i^j - q_i^{j-1})\cdot(a_i^{j-1} - x_i)}{(a_i^j - a_i^{j-1})^2} \right\} = \frac{(q_i^j - q_i^{j-1})}{(a_i^{j-1} - a_i^j)}$

Inferior limit of F_i^{j+1}: $\lim_{x_i \to a_i^{j+}} \frac{\partial Q(x,u)}{\partial a_i^j} = \lim_{x_i \to a_i^{j+}} \left\{ \frac{(q_i^j - q_i^{j+1})\cdot(a_i^{j+1} - x_i)}{(a_i^j - a_i^{j+1})^2} \right\} = \frac{(q_i^j - q_i^{j+1})}{(a_i^{j+1} - a_i^j)}$

Modal point of F_i^j : $\left.\frac{\partial Q(x,u)}{\partial a_i^j}\right|_{x=a_i^j} = 0$

In order to solve this problem of discontinuity, we impose the following constraint:

$$\lim_{x_i \to a_i^j} \frac{\partial Q(x,u)}{\partial a_i^j} = 0$$

In this case the modal point is not adjusted and it keeps its old value $a_i^j(t+1) = a_i^j(t)$. This constraint doesn't really affect the performances of the tuning algorithm because the parameters of the FPID are sufficiently close to the optimum.

2. The calculation of the $\frac{\partial Q(x,u)}{\partial a_i^j}$ becomes indefinite when $a_i^j = a_i^{j-1}$ and $a_i^j = a_i^{j+1}$.

   We can ovoid this case by imposing constraints of non overlappings of the membership functions to keep the legibility of the fuzzy rules

3. In order to insure the legibility of the fuzzy rules ovoiding labels inversions of the inputs membership functions, we imposed constraints during the adjustment process. Therefore we have to fix a superior limit and an inferior limit for each mobile modal points as can be shown in Fig. 6.

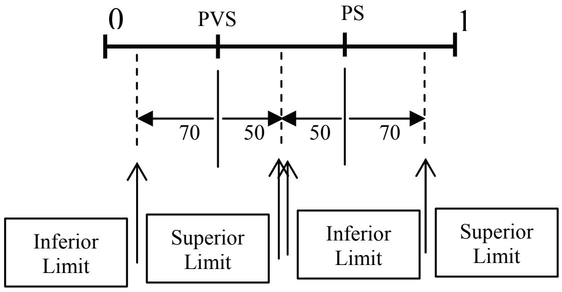


Fig. 6. Fixation of the variation limits modal points

The modal points of PS and PVS are mobile, so we have fixed their limits in order to have a maximal movement field which explains the choice of percentage (70%) to the right and to the left of respectively PS and of PVS .The field between the two modal points is equally shared (50%). The modal points of the negative part (NS et NVS) are obtained by symetry.

2-2 ***Learning algorithm for consequents parameters***

The constant coefficient of the consequent part of the fuzzy rule is also updated by a rise gradient algorithm:

$$K_i^l(t+1) = K_i^l(t) + \beta \frac{\partial Q(x_{t+1}, u_{t+1})}{\partial K_i^l} \tag{33}$$

With

$$\frac{\partial Q(x_{t+1}, u_{t+1})}{\partial K_i^l} = \frac{\partial Q(x_{t+1}, u_{t+1})}{\partial u} \cdot \frac{\partial u}{\partial K_i^l} \tag{34}$$

Not having a mathematical model of the process, we approximate the Jacobian $\frac{\partial Q(x,u)}{\partial u}$ as follow:

$$\frac{\partial Q(x_{t+1}, u_{t+1})}{\partial u} = \frac{Q(x_{t+1}, u_{t+1}) - Q(x_t, u_t)}{u_{t+1} - u_t} \tag{35}$$

The previous expression becomes very large if $u_t \approx u_{t-1}$ which can cause an inflation of the FPID parameters. To remedy it, we adopt an important and frequently used simplification which consists in replacing the Jacobian by its sign [86]. Finaly, we obtain the following equation:

$$\frac{\partial Q(x_{t+1}, u_{t+1})}{\partial K_i^l} = sign\left(\frac{\partial Q(x_{t+1}, u_{t+1})}{\partial u}\right) \cdot \alpha^l x_i \quad \text{with } x_0 = 1; \tag{36}$$

2.3. ***Execution Procedure***

We present the detailed of an one-time-step global execution procedure as follows. Let *t+1* be the current time step; the FPID obtain the new control output u_{t+1} based on (25) by using perception $\alpha_t\left(x_t\right)$. The action u_{t+1} after its transformation by the SAM becomes the $u_{c,t+1}$ (27). The action $u_{c,t+1}$ is applied on the system and makes it transits from the state x_t towards x_{t+1}. The performance of the FPID controller is evalueted by the signal of reinforcement r_{t+1}. The perception organ in common to the FPID and **QE_FIS, perceives** then this state x_{t+1} by the means of the firing strength of the rules $\alpha_t^l\left(x_{t+1}\right)$. Let us note that $\alpha_t^l\left(x_{t+1}\right)$ is subscripted by *t*, for it is the perception of x_{t+1} but using the perception parameters (modal points) which we adjusted to the time step *t*.

A. The one stage of the learning algorithm consists in the execution procedure of the continuous tuning of the antecedent parameters of the FPID/QE-FIS. The eight steps are as follows:

1. Estimation of the function of evaluation t-optimal of the current state that is provided by the QE_FIS;

$$Q_t^*(\underline{x}_{t+1}, u_{t+1}) = \sum_{i=1}^{Nmf_1} \sum_{j=1}^{Nmf_2} \alpha_t^{i,j}\left(x_{t+1}\right) \cdot q_t^{i,j} \tag{37}$$

   The indice *t* indicates that we have used the parameters adjusted to the previous time step.

2. Compute the TD error $\tilde{\varepsilon}_{t+1}$ using (21);
3. Tune parameter vector *q* of the QE_FIS according to (22) and (17);
4. Tune the antecedent parameters (modal points) of the FPID as defined in (32**)**. The term of adjustment $\dfrac{\partial Q\left(x,u\right)}{\partial a_i^j}$ is given by table 1 and the learning rate *ρ* is also adaptive according to the rule Delta Bar Delta seen in equation (17);
5. Recompute the perception since we adjusted the position of the modal points which makes change perception ;

$$\alpha_{t+1}^{i,j}(x_{t+1}) = \mu^{F_1^i}\left(x_1(t+1)\right) \cdot \mu^{F_2^j}\left(x_2(t+1)\right) \tag{38}$$

$$i = 1, \ldots, Nmf_1; \quad j = 1, \ldots, Nmf_2$$

6. Update the eligibility trace used for the parameters tuning of the QE_FIS according to (23);
7. New calculation of the evaluation function corresponding to the action (u_{t+1}) and the current state (x_{t+1}) with the new parameters ;

$$Q_{t+1}(\underline{x}_{t+1}, u_t) = \sum_{i=1}^{Nmf_1} \sum_{j=1}^{Nmf_2} \alpha_{t+1}^{i,j}\left(x_{t+1}\right) \cdot q_{t+1}^{i,j} \tag{39}$$

   This value will be used for the calculation of the TD error at the next step of time.

8. Calculation of the control ouput u_{t+1}, by using new perception;

$$u_{t+2} = \sum_{i=1}^{Nmf_1} \sum_{j=1}^{Nmf_2} \alpha_{t+1}^{i,j}\left(x_{t+1}\right) \cdot u^{i,j} \tag{40}$$

B. The stage seconde of the learning algorithm consists in the execution procedure of the continuous tuning of the consequent parameters of the FPID. The antecedent parameters are fixed to the values found in the stage one The sevent steps are as follows:

1. Estimation of the function of evaluation t-optimal of the current state $Q_t^*(\underline{x}_{t+1}, u_{t+1})$ as defined in (38) , that is provided by the QE_FIS ;
2. Compute the TD error $\tilde{\varepsilon}_{t+1}$ using (21);
3. Tune parameter vector *q* of the QE_FIS according to (22) and (17);
4. New calculation of the t-optimal evaluation function of the current state by using the new parameters of the QE_FIS:

$$Q_{t+1}(\underline{x}_{t+1}, u_{t+1}) = \sum_{i=1}^{Nmf_1} \sum_{j=1}^{Nmf_2} \alpha^{i,j}\left(x_{t+1}\right) \cdot q_{t+1}^{i,j} \tag{41}$$

5. Update the eligibility trace used for the parameters tuning of the QE_FIS according to (23) ; This stage completes the learning part of the QE_FIS
6. Tune the parameters of the FPID according to (33) and (36) ;

We use perception $\alpha^l\left(x_t\right)$ and $x_j(t)$ for the tuning parameters of FPID, for it is them which are responsible for the issue of the control ouput u_{t+1}

7. Calculation of the control ouput u_{t+2}, by using the new parameters of FPID:

$$u_{t+2} = \sum_{i=1}^{Nmf_1} \sum_{j=1}^{Nmf_2} \alpha^{i,j}\left(x_{t+1}\right) \cdot u_{t+1}^{i,j} \tag{42}$$

6. Simulations-experimentations

The proposed fuzzy reinforcement-learning controller has been simulated for the DC/DC buck converter control problem and some simulation and experimentation results are presented.

1. DC/DC Buck Converter

The system is a 1kW DC/DC Buck Converter with the control of its output voltage in current mode which consists in imposing the current peak which crosses the smoothing inductance L therefore also the output current in the variable resistive load. It has an impulse response under deadened. We will carry out a voltage fuzzy control of the cascade type. To restrict the influence of measure noise, the output voltage is filtered using a 50 Hz low-pass first-order filter. The general diagram of such a chopper is given in fig.7**.** The values of the different components of the system are given by table 2.

The inner structure of the FPID part and its Fifteen (or twenty six) resulting tuning factors are defined on the normalised universe of discourse by the user. Thus values for the scaling factors *dem* and *gm* and the integrator gain called *Ki* have to be defined. The scaling factor *em* is supposed to be defined considering the benchmark step magnitude. Due to their global effect on the control performance and robustness, optimal input and output scaling factors play critical role in the FPID controller and they have the highest priority in terms of tuning and optimisation [68],[79]. To tune theme a lot of strategies have been proposed. Only these last three parameters pre-established values that we will provide in this part will depend on the open-loop identification of the process and on the sample period. In this way, this methodology can be seen as a Ziegler-Nichols -like tuning strategy for fuzzy controllers. In order to evaluate the performances of the control strategy, this pre- established set will be provided for a very common industrial benchmark: no-load step response, nominal load regulation, no-load regulation. It will warrant a high performances process response on this benchmark combined with a high robustness. The non-linear charge will be connected or disconnected according to the following functioning cycle fixed for the whole of our tests at 0.5 second: the no-load starting (R_0 = 20Ω) is carried out at t=0s; to sudden maximum load connection (R = 150Ω) at t=0.166s (at the third of the horizon); and sudden disconnection (R_0 = 20Ω) intervenes at t=0.333s (at the two thirds of the horizon).

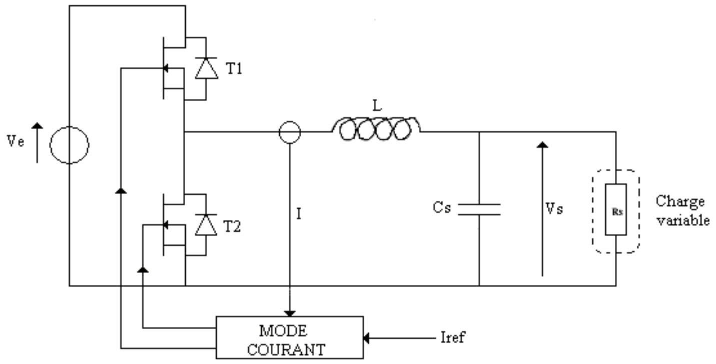


Fig. 7. General diagram of the Buck chopper

V_e	60V	α_{min}	0.05
L	1.3 mH	R_{load}	20 Ω
C_s	165 μF	R_0	150 Ω
I	≤10A	V_s	0 to 10 V

TABLE II. System Parameters Values

2.Open-loop identification test

In this paper, in the first part we want to propose pre-established settings for this FPID that are based on just one open-loop step identification test of the process.These predefined settings already exist in case of use of the FLC with an open-loop stable or evolutive process [62-63], that are widely found in electrotechnical applications. The control quality for this system will be evaluated considering the Integrative Absolute Error (IAE) between the reference and the measured signal on the specified benchmark. The FPID will then be all the more robust since these IAE-criterion remains insensitive to uncontrolable factors (white noise, process misindentification ...).

The open-loop identification of this stable process like DC/DC Buck converter will provide three parameters called *K*, *T* and τ for an open-loop transfer function of the process that can be expressed as Eq. 4. Figure 8 presents the open-loop response of the considered system.

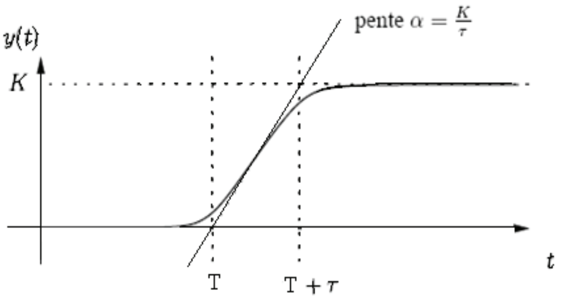


Fig. 8. Open-loop response of the considered system

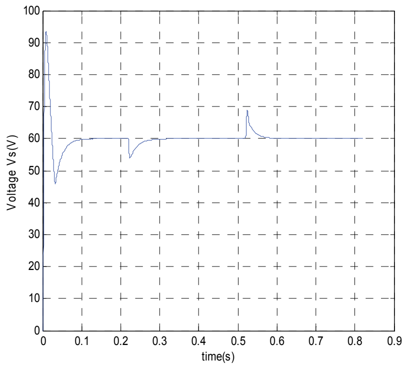
$$FT(p) == K.\frac{e^{-Tp}}{1+\tau p} = FT^{H}_{50Hz} = 14.7\cdot\frac{e^{-0.0028p}}{1+0.0174p} \tag{43}$$

with: K=14.7, T=0.0028s, τ=0.002.8s
Pre -established settings for the FPID-like FLC is provided in Table 3. The choice is inspired from the Broïda predefined settings [62] for conventional PID controllers by imposing an equal distribution of the membership functions of the inputs/outputs variables and in the case of a value step magnitude c with Tech the sampling period equal to 0.00015 s.

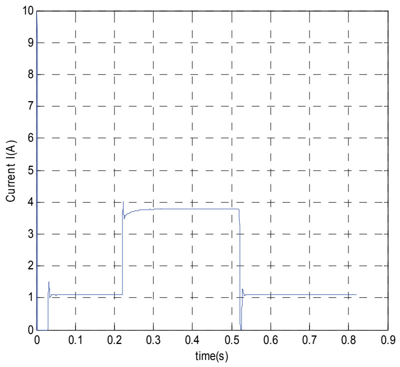
Factors	Level
PS_e	0.67
PVS_e	0.33
PS_{de}	0.67
PVS_{de}	0.33
PS_s	0.67
PVS_s	0.33
e_m	$c = 60V$
K_i	$K_i = \frac{0.8}{K\cdot T} = 19.4363$
de_m	$de_m = \frac{2.5}{\tau\cdot T}\cdot(\tau+0.4\cdot T)\cdot T_{ech}\cdot c = 8.5530$ dV/dt
g_m	$g_m = \frac{0.8}{K\cdot T}\cdot(\tau+0.4\cdot T)\cdot c = 21.5977$

TABLE III. Pre-Defined Factors Levels

Fig. 9 and Fig. 10 give the simulation results for the DC/DC converter controlled either by a conventional PID controller whose parameters have been set according to the Broida settings or by our FPID with pre-established settings (table 2) i.e. equally -distributed for antecedent memberships function and consequent values. With the FPID, all the simulations lead to an outstanding improvement of the behavior with respect to the standard tunings of the Broida PID controller. The measured IAE-criterion is 0.9729 V.s in the FPID case and 1.134 V.s in the classical PID case. Beyond this 17%-criterion improvement, we can underline the similar behavior of the two controllers.



Output voltage control



Imposed reference current

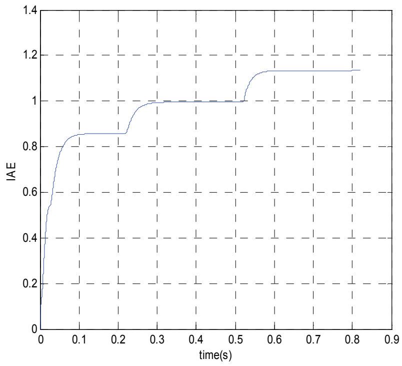
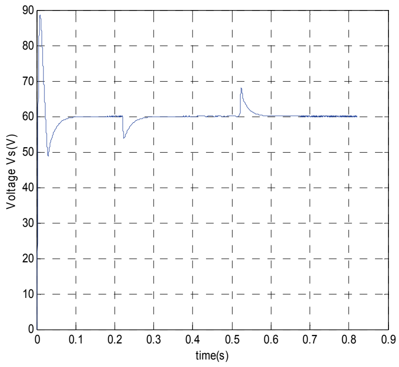


Fig. 9. Broida PID controller

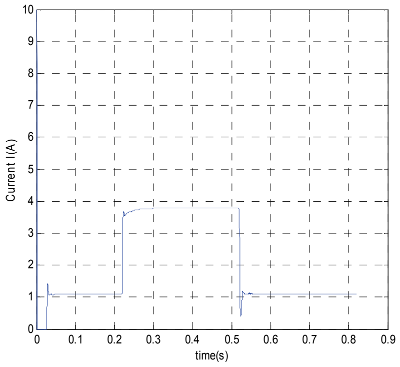
3. Discret parameter tuning

Now, we apply the discret FQL algorithms to tuning the antecedent parameters then the consequent parameters of the FPID controller determined previously. The values for the scaling factors *dem, gm, Ki* and *em* are fixed during the learning processus and therefore we have eight parameters tuning (table 3). This doesn't constitute a difficulty because the reinforcement learning method doesn't depend on the parameter number to tune , which constitute a considerable advantage as compared to the other tuning methods such as the experimental product-plan method [62],[63].

First of all, it's necessary to define the levels (two in our case) for each of the considered controlable factors. Referring to Figure 2 and seeing the 50%-overlapping rate as a mechanical constraint, the positions of the PVS's and PS's membership functions apex aren't totally independent. When the position of PS is fixed at x (x between 0 and +1), the position of PVS can only be between 0 and x. Thus, it's more advisable to define the two levels of these input factors relatively [62]. Under his assumption, the two chosen levels for each of the eight tuning factors are provided in Table 4.



Output voltage control



Imposed reference current

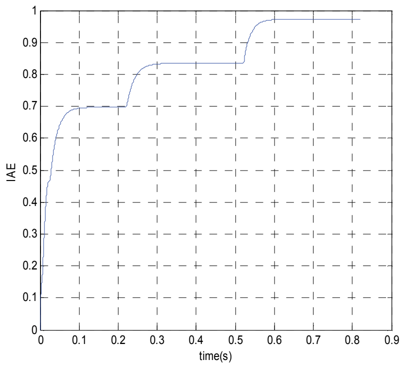


Fig. 10. Broida FPID controller

Factors	Level 1	Level 2	Level 3	Level 4
PS_e	0.30	0.40	0.60	0.75
PVS_e	0.12	0.17	0.23	0.28
PS_{de}	0.30	0.40	0.60	0.75
PVS_{de}	0.12	0.17	0.23	0.28
PS_s	0.20	0.40	0.60	0.80
PVS_s	0.20	0.40	0.60	0.80
NS_s	-0.20	-0.40	-0.60	-0.80
NVS_s	-0.20	-0.40	-0.60	-0.80

TABLE IV. Controlable Factors Levels

The choice of these levels has been achieved so a to sweep the area of researches.

3.1 Reinforcement function

The next step is to define the reinforcement signal *r*. The process performances are evaluated by the maximum absolute value of the error deviation *e*, because the objective of the FPID controller is to minimise this error as small as possible in tracking and in regulation. The reinforcement signal *r* may be defined as follows:

$$r(t) = \begin{cases} -\dfrac{|e|}{e_{\max}} & If \quad |e| \le e_{\max}, \text{success} \\ -1 & If \quad |e| \succ e_{\max}, \text{failure} \end{cases} \tag{46}$$

We adopt a reinforcement function with a punitive policy in order to obtain optimal parameters in a short time period. In our simulations, we assume the maximum allowed error as 0.05. The parameters used in the simulations and experimentations are summarized in Table 5. The second value of parameter indicated in this table relates to the learning algorithm for consequents parameters.

Parametres	Symbol	Value
Discount factors	γ	0.9/0.95
Proximity factors of the omission function (eligibility traces)	λ	0.9
Initial learning rate	ρ	0.1
Interferer in the adaptative learning rate	K_ρ	0.0001/ 0.001
Interferer in the adaptative learning rate	ψ_ρ	0.5/ 0.001
Interferer in the adaptative learning rate	φ_ρ	0.5
Tolerance Noise/Amplitude of the quality	S_p	0.1
Proportioning of the exploration	θ	1

TABLE V. Parameters of the Discrete FQL algorithm

At first we tried to put into functions the two algorithms (FQL applied to the premises and FQL applied to the conclusions) similarly. But different attempts have shown that a simultaneous tuning of antecedent and consequent parameters lead to instability, therefore we opted for a separated tuning procedure beginning with the FQL for antecedent part followed by those of the consequent part.

The final tunings are provided in Table 6. Figures 11, 12 and 13 present the simulations results on the DC/DC converter. The "before optimisation" results have been obtained when the membership functions and the singletons are equidistributed on the universe of discourse. This kind of "medium"setting can be seen as a preliminary setting but constitutes unfortunately often the final setting of the FLC too.

3.2 Simulations and experimental results

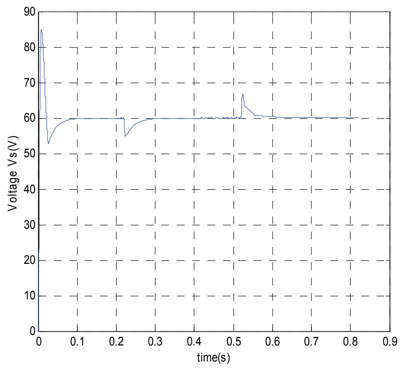
Stage one: antecedent parameters tuning

We use the FQL algorithm applied to the antecedents with equally -distributed consequent values. We obtain the following results.

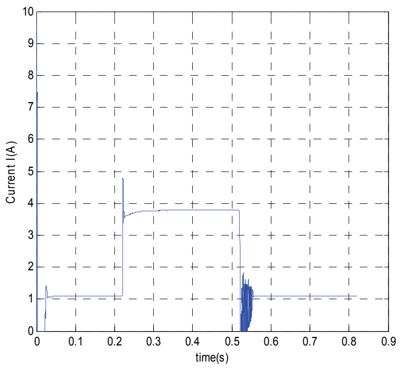
Factors	PS_e	PVS_e	PS_{de}	PVS_{de}
Level selected	0.75	0.12	0.3	0.17

TABLE VI. Modal points determined by discrete tuning

The results of simulations (fig.11) during the learning stage give an IAE criterion equals to 0.8544. We can notice then an improvement due to a larger space of solutions.



Output voltage control



Imposed Reference Current

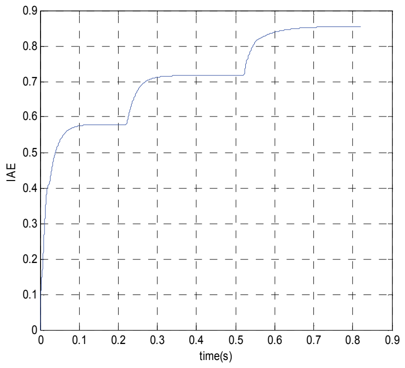
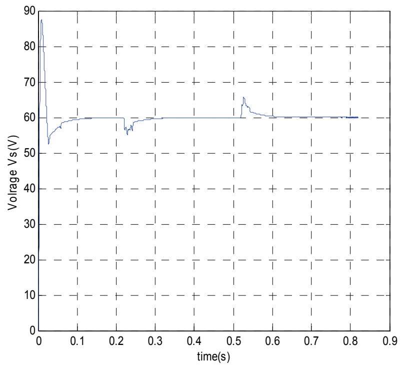


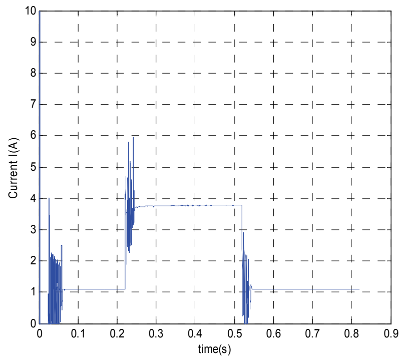
Fig.11. tecedent tuning of the FPID controller

Stage second: consequent parameters tuning

Initially, to show the importance to tune the antecedents parameters, we execute classic algorithm FQL with equally -distributed for antecedent memberships function. The results of simulations obtained during the learning stage give an IAE criterion equals 0.938, what corresponds to a deterioration of 10% of the optimization criterion as compared to the value obtained with classic FQL algorithm. Then we use the consequent FQL algorithm with the modal point values of the input membership functions determined in stage one.We get then the rule base showned in the table 7. The results of the simulations obtained during the learning (fig 12 and 13) show an improvement of the IAE criterion, equals now to 0.5318 compared to respectevely 0.8544 for the during the learning and validation phase. Therefore we have reached an improvement of the criterion of 12% as compared to the stage one**.** These simulations results show an important decreasing of the optimisation criterion (about 45%) as compared to the value obtained before the tuning by FQL algorithms. More over the behaviour of tracking is improved because the response time is shorter.



Output voltage control



Imposed rference current

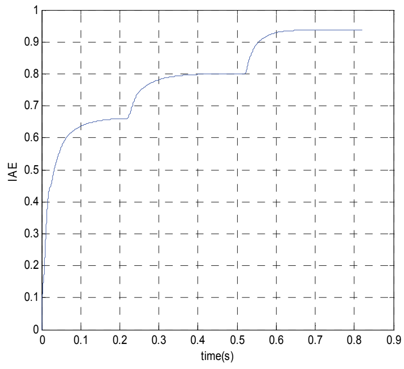
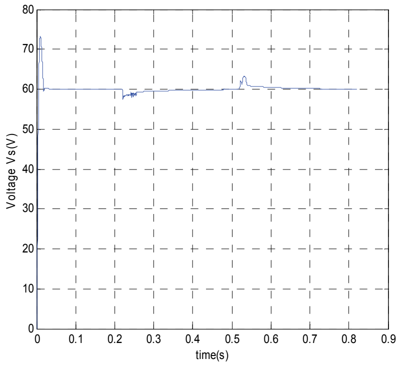


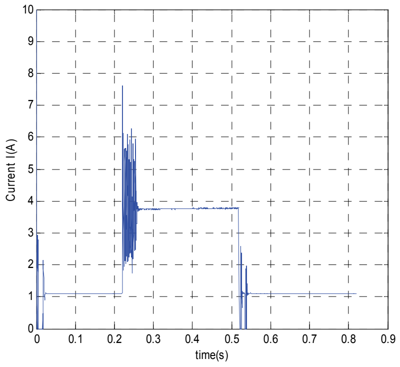
Fig. 12. Consequent tuning of the FPID controller with classic FQL algorithm

		de						
		NB	NS	NVS	Z	PVS	PS	PB
e	NB	-1	-1	-1	-1	-0.2	-0.8	0
	NS	-1	-1	-1	-0.4	-0.4	0	0.8
	NVS	-1	-1	-0.8	-0.6	0	0.4	0.2
	Z	-1	-02	-0.8	0	0.6	0.4	1
	PVS	-0.8	-0.8	0	0.8	0.8	1	1
	PS	-0.4	0	0.8	0.2	1	1	1
	PB	0	0.4	0.8	1	1	1	1

TABLE VII Antidiagonal rule base



Output voltage control



Imposed rference current

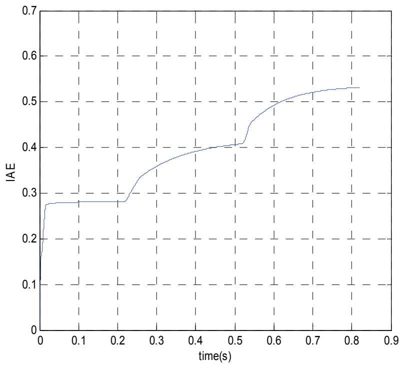


Fig.13. Consequent tuning of the FPID controller (table 6 with antidiagonal rules table 7)

4. Continuous parameter tuning

The previous obtained results (tables 6 and 7) constitute values close to the optimum which just need to be refined by means of the proposed continuous tuning algorithm. The parameters of the continuous FQL algorithm used in the simulations/experimentations are summarized in Table 8.

The simulation results obtained with non antidiagonal table 10 shown on system responses of the fig.14 , an IAE criterion of 0.4789 during the learning phase, which constitute a decreasing of 10% in comparison to the discreet tuning and of 58% in comparison to the classic Broïda PID. Let's remind also that the method of experimentation plans measure an IAE criterion of 1.05 (standard tuning) and 0.56 (robust tuning) with 16 experimental tests, therefore we have reached an improvement of 54% (standard tuning) and 15% (robust tuning) in relation to this method.

Parametres	Symbol	Value
Discounted factor	γ	0.9
Proximity factor of the omission function (eligibility traces)	λ	0.9
Learning rate QE_FIS	ρ	0.0001
Interfere in the adaptative learning rate	K_η	0.00001
Interfere in the adaptative learning rate	ψ_η	0.5
Interfere in the adaptative learning rate	φ_η	0.5
Learning rate FPID control	β	0.00001
Learning rate of the antecedents	ρ	0.00001

TABLE VIII Parameters of the Continuous FQL algorithm

We obtain the following results with the continuous FQL algorithm.

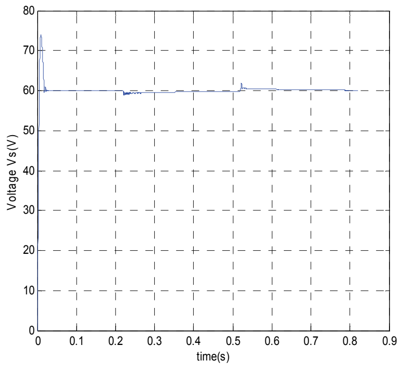
Factors	PS_e	PVS_e	PS_{de}	PVS_{de}
Level selected	0.75	0.07	0.3	0.171

TABLE IX Modal points determined by continuous tuning

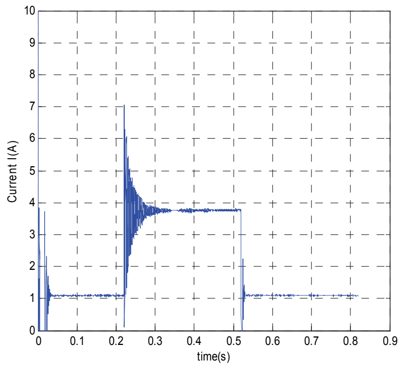
		de						
		NB	NS	NVS	Z	PVS	PS	PB
e	NB	-1	-1	-1	-1	-0.2	-0.8	0
	NS	-1	-1	-1	-0.4001	-0.4	0	0.6
	NVS	-1	-1	-0.8	-0.6006	0	0.4	0.4
	Z	-1	-0.2	-0.8	0	0.6	0.4	1
	PVS	-0.8	-0.8	0	0.8005	0.8	1	1
	PS	-0.4	0	0.8	0.2	1	1	1
	PB	0	0.4	0.8	1	1	1	1

TABLE X Non Antidiagonal rules base

The best values controllable factors determined by simulations are then used to carry out the experimental tests.The experimental results (fig 16) are very similar to the simulation results, which confirms the performances of experimental methodology proposed. Moreover, it has better results on the the system responses and IAE criterion in comparison with standard and robust methods experimental plans as can be seen in Fig.15. The Table 11 relating to the IAE criterion confirmed the quality of the obtained responses by our approach. The different results show that it carries out one compromised for the two types of operation in tracking and in regulation. Also, the performances of the system can be greatly improved by executing a specific FPID controller for each operation type in comparison to one FPID controller functioning on all the operating range.The figure 17 shows that the time of response and the time of reject of the perturbations have considerably decreased. These simulations results show an important decreasing of the optimisation criterion (about 2%) as compared to the value obtained with the only using of one FPID controller tuning by FQL algorithms.



Output voltage control



Imposed reference current

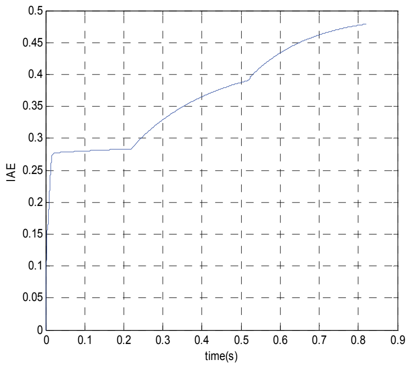
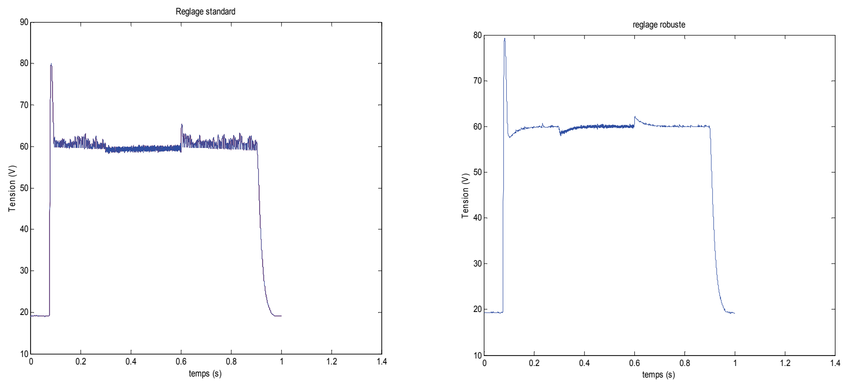
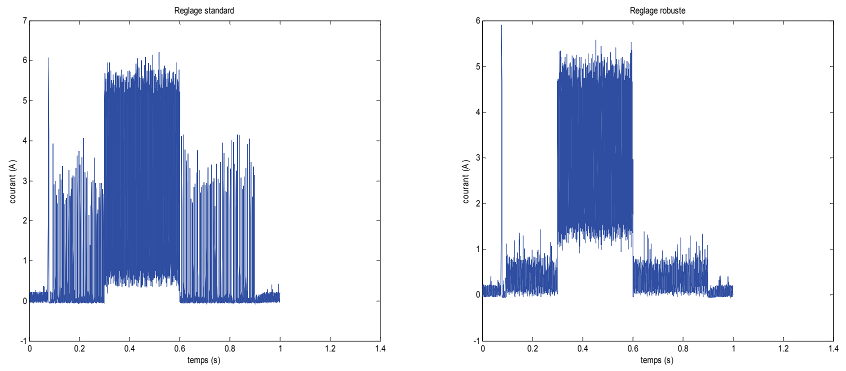


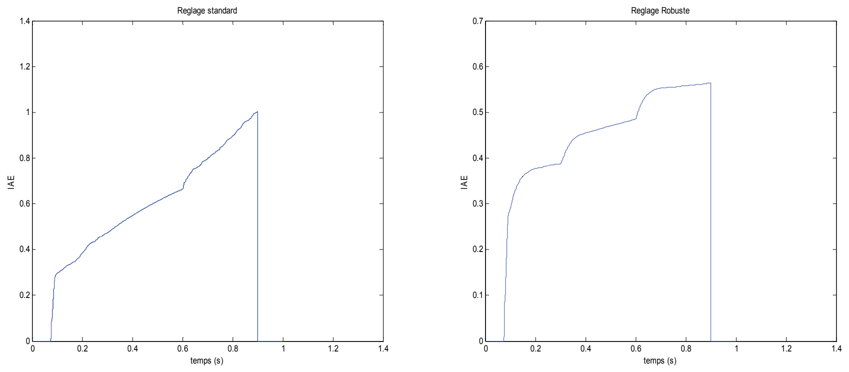
Fig. 14. Continuous tuning of the FPID controller



Output voltage control

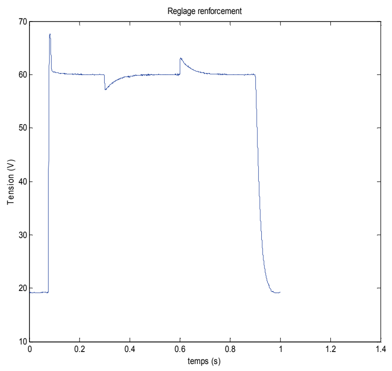


Imposed reference current

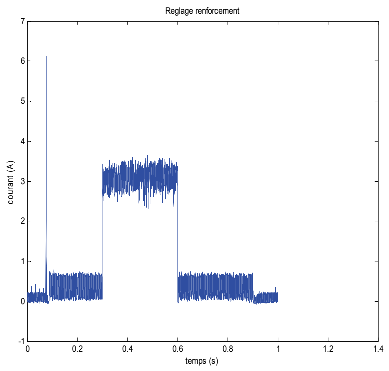


IAE criterion

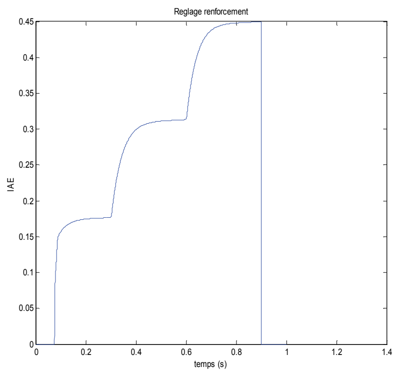
Fig.15. Experimental results of FPID controller (standard and robust methods of experimental plans)



Output voltage control



Imposed reference current

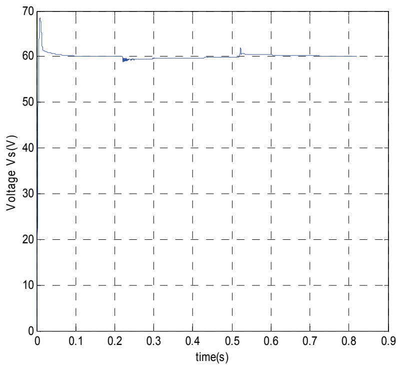


IAE criterion

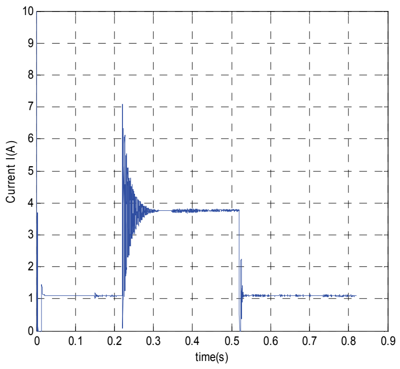
Fig.16. Experimental results of the FPID controller continuous tuning

Methods	IAE
PID Broida	1.134
FPID Broida	0.9729
Classic FQL	0.9
Discret FQL	0.5454
Continous FQL (one FPID),	0.5396
Continous FQL (two FPID)	0.4693
Expermental plans (standard)	1.05
Expermental plans (robust)	0.56

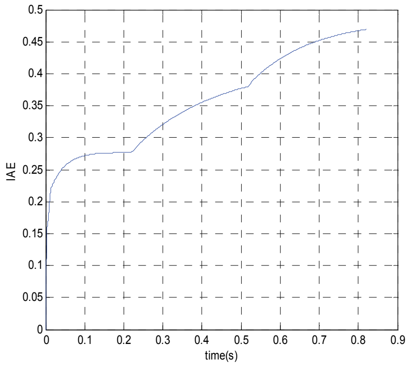
TABLE XI Comparative table of the various methods



Output voltage control



Imposed reference current



IAE criterion

Fig.17. Simulation results ofcommutation between two FPID controller obtained by method continuous tuning

7. Conclusion

On the whole, this paper provides some new and original headlines for the on-site tuning PID like fuzzy logic controllers.With the Broida methodology for conventional PID controllers; it is possible to obtain satisfying basic FPID. These settings depend only on two or three parameters (K, T and eventually τ) extracting from an open-loop identification test of the process.

A second set of pre-defined settings for the controllable factors FPID is proposed and selected from the reinforcement learning design procedure. Two discrete tuning algorithms are developed based FQL for antecedent parameters and consequent parameters tuning respectively. The process performances are always evaluated considering the IAE-criterion between the reference and the measured signal. The FPID will then be all the more robust since this IAE-criterion remains insensitive to uncontrollable factors (here white noise, process misidentification or high order process). This second stage tuning could be seen as an on-site roughly FPID tuning strategy.

Finally the paper presents the controller architecture composed of the QE-FIS and the FPID for solving continuous space reinforcement learning problems and fine tuning controllable factors. The QE-FIS is used to estimate the optimal action-value function, and the FPID is used to get the control output based on the estimated action value function provided by the QE-FIS. With the proposed architecture, the parameters learning algorithms for the QE-FIS and the FPID are developed based on techniques of temporal difference and rise gradient algorithm. The two stages (discrete tuning and continuous tuning) complete one another because the first stage does a preliminary tuning which reduces the field of researches and provides an initialization to the second stage.This initialization of the parameter vector close to the optimum enables to ovoid the local optimums of the gradient methods. Finally, the simulations and the experimentation of the DC/DC converter demonstrate the validity and performance of the proposed learning algorithms and give better results as compared to the method of experimentation plans.

8. Acknowledgment

Hacène Rezine: was born in 1955 in El-Kala, Algeria. He received an engineering degree from the Ecole Nationale d'Ingénieurs et de Techniciens, Alger in 1979, the Engineer Degree from the Ecole Nationale Supérieur d'Electrotechnique, d'Electronique, d'Informatique, et d'Hydraulique, Toulouse, France in 1980 and the Doctor-Engineer Degree from the Institut Nationale Polytechnique of Toulouse in 1983. He is currently the Head of Unit of teaching and control research of the Algerian Polytechnic Military School. His current research areas include Electrical drives, Mobile robotic, Fuzzy control and Artificial Intelligent Control.
Rabah Louali: was born in October, 15th, 1982 in Blida (Algeria). He received the Engineering degree in 2006, from the Military Polytechnic School (Algeria). Currently, he is Magister student at this school. His current research areas include Mobile Robotic, Control of System and Fuzzy control.

9. References

[1] M. Fei, SLHo, "Progress in On-line Adaptive, Learning and Evolutionary Strategies for Fuzzy Logic Control", IEEE 1999 International Conference on Power Electronics and Drive Systems, Hong Kong. , pp 1108-1113. July 1999]
[2] Chin-Teng Lin, '' A neural fuzzy control system with structure and parameter learning'', Fuzzy Ssets and Systems, (70), pp 183-212, 1995
[3] F. Herrera, M. Lozano & J.L. Verdegay, "Tuning fuzzy logic controllers by genetic algorithms," Int. J. Approximate reasoning, v. 12, pp 299-315, 1995.
[4] Meng Joo Er, Member, IEEE, and Chang Deng, "Online Tuning of Fuzzy Inference Systems Using Dynamic Fuzzy Q-Learning ", IEEE Transactions on Systems, Man, and Cybernetics—PartT B: Cybernetics, Vol. 34, No. 3, June 2004
[5] L. Jouffe: "Actor-Critic Learning Based on Fuzzy Inference System", Proc of the IEEE Conference on Systems1996. Man and Cybernetics. Beijing, China, pp.339-344, 1996
[6] C. Watkins, P. Dayan: "Q-Learning Machine Learning", pp.279-292, 1992.
[7] P. Y. Glorennec and L. Jouffe, "Fuzzy Q-learning," Proc. OfIEEE Int. Con$ On Fuzzy Systems, pp. 659-662, 1997

[8] Dongbing Gu- Huosheng Hu- Libor Spacek, ": Learning Fuzzy Logic ontroller for Reactive Robot Behaviours", Proceedings of the 2003 IEEEASME International Conference on Advanced Intelligent Mechatronics, pp 46-51, 2003.
[9] C.C. Lee, "Fuzzy logic in control systems: fuzzy logic controller- part I&II," IEEE T-SMC, v.20, n.2, pp404-435, 1992.
[10] Mitra, S., Hayashi, Y. (2000). Neuro-Fuzzy Rule Generation: Survey in Soft Computing Framework, IEEE Trans. On Neural Networks, Vol. 11, No. 3...
[11] Dongbing Gu and Huosheng Hu, « Reinforcement Learning of Fuzzy Logic Controllers for Quadruped Walking Robots", Copyright 2002 IFAC 15th Triennial World Congress, Barcelona, Spain.
[12] Chia-Feng Juang, « Construction of Dynamic Fuzzy If-Then Rules through Genetic einforcement Learning for Temporal Problems Solving », IEEE, pp 2341-2346, 2001.
[13] Meng Joo Er, Member, IEEE, and Chang Deng, "Online Tuning of Fuzzy Inference Systems Using Dynamic Fuzzy Q-Learning ", IEEE Transactions On Systems, Man, And Cybernetics—PartT B: Cybernetics, Vol. 34, No. 3, June 2004.
[14] Jang J. C, ''. ANFIS: Adaptive-Neural-Based Fuzzy Inference System'', IEEE Trans. on SMC, Vol. 23, pages 665- 685, Aug. 1993.
[15] J. S. R. Jang, C. T. Sun, and E. Mizutani, Neuro-Fuzzy and Soft Computing.Englewood Cliffs, NJ: Prentice-Hall, 1997.
[16] S. Wu, M. J. Er, and Y. Gao, "A fast approach for automatic generation of fuzzy rules by generalized dynamic fuzzy neural networks," IEEE Trans. Fuzzy Syst., vol. 9, pp. 578–594, Aug. 2001.].
[17] C. W. Anderson, "Strategy learning with multilayer connectionist representations," in Proc. 4th Int. Workshop on Mach. Learn., Irvine, CA, June 1987, pp. 103–114.
[18] Bonarini, A. Evolutionary Learning of Fuzzy rules:competition and cooperation, In Pedrycz, W. (Ed.), FuzzyModelling: Paradigms and Practice, Kluwer Academic Press,Norwell, MA, pp265 – 284,1997.
[19] Juang, C. F., Lin, J.Y., and Lin, C. T. (2000). Genetic Reinforcement learning through Symbiotic Evolution for Fuzzy Controller Design, IEEE Transactions on SMC-Part B, Vol. 30, No. 2, pages 290-301.
[20] C. Karr, "Genetic algorithms for fuzzy controllers," AI Expert, vol. 2, pp. 27–33, 1991.
[21] M. Lee and H. Takagi, "Integrating design stages of fuzzy systems using genetic algorithm," in Proc. 2nd IEEE Int. Conf. Fuzzy Systems, San Francisco, CA, 1993, pp. 612–617.
[22] K. Kropp, "Optimization of fuzzy logic controller inference rules using a genetic algorithm," in Proc. EUFIT'93, Aachen, Germany, 1993, pp.1090–1096.
[23] T. Kawabe, T. Tagami, and T. Katayama, "A genetic algorithm based minimax optimal design of robust I-PD controller," in Proc. IEE Int.Conf. Control, London, U.K., 1996, pp. 436–441.
[24] Berenji, H. R., Khedkar, P. (1992). Learning and Tuning Fuzzy Logic Controller through Reinforcements, IEEE Trans. OnNeural Networks, Vol. 3, No. 5, pages 724-740. September, 1992.
[25] Chin-Teng Ling, C. S. George Lee, "Reinforcement Structure/Parameter Learning for Neural-Network- Based Fuzzy Logic Control Systems", IEEE Trans. on Fuzzy Systems, vol. 2, no. 1, Feb. , 1994.

[26] C. T. Lin and C. S. G. Lee, "Reinforcement structure/parameter learning for an integrated fuzzy neural network," IEEE Trans. Fuzzy Syst., vol. 2, pp. 46–63, Feb. 1994

[27] Kaelbling, L. P., Moor, A. W. (1996). Reinforcement Learning: A Survey, Journal of Artificial Intelligence Research, Vol. 4, pages 237-285

[28] C. K. Chiang, H. Y. Chung, and J. J. Lin, "A self-learning fuzzy logic controller using genetic algorithms with reinforcements," IEEE Trans. Fuzzy Syst., vol. 5, no. 3, pp. 460–467, Jun. 1997.

[29] C. T. Lin and C. P. Jou, "Controlling chaos by GA-based reinforcement learning neural network," IEEE Trans. Neural Networks, vol. 10, no. 4,pp. 846–859, Jul. 1999.

[30] Lin, C. T., C. P. Jou,. " GA-Based Fuzzy Reinforcement Learning for Control of a Magnetic Bearing System", IEEETrans. On SMC-Part B, Vol. 30, No. 2, pages 276-289, 2000.

[31] Hyo-Byung Jun and Kwee-Bo Sim, "Behavior Learning and Evolution of Collective Autonomous Mobile Robotsbased on Reinforcement Learning and Distributed Genetic Algorithms",IEEE International Workshop on Robot and Human Communication 1997 IEEE,pp 248,253

[32] A. G. Barto and M. I. Jordan, "Gradient following without backpropagation in layered networks," in Proc. of IEEE I" Annual Con$ Neural Networks, vol. 2, San Diego, CA, 1987, pp. 629-636

[33] C. W. Anderson, ''Learning and Problem Solving With Multilayer Connectionist Systems '' , Ph.D. disseration, Univ. Massachusetts, Amherst, 1986.

[34] Kaelbling, Littman, and Moore (1996)[Kaelbling, L. P., Littman, M. L., & Moore, A. W. (1996). Reinforcement learning: A survey.Journal of Arti_cial Intelligence Research, 4, 237{285.

[35] Sutton, R. S., & Barto, A. (1998). Reinforcement Learning: An Introduction. MIT Press, Cambridge, MA

[36][David E Moriarty- Alan C.Schultz-John J.Grefenstette, "Evolutionary Algorithms for Reinforcement Learning", Journal of Arti_cial Intelligence Research 11 (1999) 241-276 .

[37] R. S. Sutton, "Learning to predict by the methods of temporal differences," Mach. Learn., vol. 3, no. 1, pp. 9–44, 1988.

[38] R. S. Sutton and A. G. Barto, Reinforcement Learning. Cambridge, MA: MIT Press, 1998.

[39] J. N. Tsitsiklis and B. V. Roy, "An analysis of temporal-difference learning with function approximation," IEEE Trans. Autom. Control, vol. 42, no. 5, pp. 674–690, May 1997

[40] A. G. Barto, R. S. Sutton, and C. W. Anderson, "Neuron like adaptive elements that can solve difficult learning control problems," IEEE Trans. Systems, Man, Cybern., vol. SMC-13, no. 5, pp. 834–846, Sep. 1983.

[41] C.W. Anderson, "Learning to control an inverted pendulum using neural networks," IEEE Control Syst. Mag., vol. 9, pp. 31–37, 1989.

[42] C. J. Wakins, "Learning from delayed rewards," Ph.D. dissertation, King's College, Cambridge, U.K., 1989.

[43] C. J.Wakins and P. Dayan, "Technical note: Q-learning," Mach. Learn., vol. 8, pp. 279–292, 1992.

[44] C.W. Anderson, "Learning to control an inverted pendulum using neural networks," IEEE Control Syst. Mag., vol. 9, pp. 31–37, 1989

[45] Chia-Feng Juang, « Combination of Online Clustering and Q-Value Based GA for Reinforcement Fuzzy System Design", IEEE Transactions On Fuzzy Systems, Vol. 13, No. 3, June 2005, pp 289-302.
[46] H. R. Berenji, "Fuzzy Q-learning for generalization of reinforcement,"in Proc. IEEE Int. Conf. Fuzzy Systems, 1996, pp. 2208–2214.
[47] Rummery, G. A., On-Line Q-Learning Using Connectionist Systems, Tech. Rep., CUED/F-INFENG/TR 166, CambridgeUniversity, 1994.
[48] P. Y. Glorennec, "Fuzzy Q-learning and dynamic fuzzy Q-learning," in Proc. IEEE Int. Conf. Fuzzy Systems, vol. 1, Orlando, FL, 1994, pp. 474–479.
[49] T. Horiuchi, A. Fujino, O. Katai, and T. Sawaragi, "Fuzzy interpolationbased Q-learning with continuous states and actions," in Proc. IEEE Int. Conf. Fuzzy Systems, 1996, pp. 594–600.
[50] P. Y. Glorennec and L. Jouffe, "Fuzzy Q-learning," in Proc. IEEE Int. Conf. Fuzzy Systems, 1997, pp. 659–662.
[51] L. Jouffe, "Fuzzy inference system learning by reinforcement methods," IEEE Trans. Syst., Man, Cybern. C, Appl. Rev., vol. 28, no. 3, pp. 338–355, Aug. 1998.
[52] M. S. Kim and J. J. Lee, "Constructing a fuzzy logic controller using evolutionary Q-learning," in Proc. 26th Annu. Conf. Industrial Electronics Society, 2000, pp. 1785–1790
[53] R.Ssutton, ''Integrated Architecture for Learning, Planing, and Reacting Based on Approximating Dynamic programming'', Proc.of 7th international Conference on machine Learning, pp 216-224, 1990
[54] R.A Mccallum, ''Using Transition Proximity for Faster Reinforcement Learning, Proc.of 9th international Conference on machine Learning, pp 316-321, 1992
[55] S.P.Singh, ''Transfert of learning by composing solutions of elemental sequential tasks, Proc.of 7th international Conference on machine Learning, pp 323-339, 1992
[56] J.Peng, "Efficient memory-based Dynamic programming, Proc.of 12th international Conference on machine Learning, pp 438-446, 1995
[57] P. Y. Glorennec and L. Jouffe, "A reinforcement learning method for an autonomous robot," in Proc. EUFIT'96, 4th Eur. Congr. Intell. Tech. Soft Comput. Aachen, Germany, pp. 1100–1104
[58] Xiaohui Dai, Chi-Kwong Li, « An Approach to Tune Fuzzy Controllers Based on Reinforcement Learning for Autonomous Vehicle Control", IEEE Transactions On Intelligent Transportation Systems, Vol. 6, No. 3, September 2005, pp 285-293
[59] R. S. Sutton, "Integrated architectures for learning, planning, and reacting based on approximating dynamic programming," in Proc. 7th Int. Workshop Machine Learning, 1990, pp. 216–224.
[60] L. J. Lin, "Self-improving reactive agents: Case studies of reinforcement learning frameworks," in Proc. 1st Int. Conf. Simulation of Adaptive Behavior: From Animals to Animals, 1991, pp. 297–305
[61] H. R. Berenji, "Fuzzy Q-Learning: A new approach for fuzzy dynamic programming," in Proc. IEEE Int. Conf. Fuzzy Systems, 1994, pp. 486–491
[62] D. Hissel, P. Maussion, G. Gateau, J. Faucher,"Fuzzy Logic Control Optimization of Electrical Systems using Experimental Designs", in Proceedings of the EPE'97 Conference, Trondheim, Norv_ege, pp. 1.090-1.095, 1997.

[63] P. Maussion, D. Hissel, "Optimized Fuzzy Logic Controller Parameters for Open-loop Stable or Evolutive Electromechanical Systems", in Proceedings of the IECON'98 Conference, Aachen, 1998.
[64] S. Joe Qin, '' Auto-Tuned Fuzzy Logic Control'', Proc.of the America Control Conference Baltimore, Maryland, June 1994, pp 2465-2469
[65] Yang Xinxin" Yang Lei, He Kezhong, Huang Shengle, Guo Muhe, Zhang Bo, "A Double-Level Fuzzy Controller with an Intelligently Adjusting Strategy of Quantization and Scale Factors,1996 IEEE, pp 280-285.
[66] George K. I. Mann, Bao-Gang Hu, and Raymond G. Gosine, "Analysis of Direct Action ,Fuzzy PID Controller Structures", IEEE Transactions On Systems, Man, and Cybernetics—Part b: Cybernetics, Vol. 29, No. 3, June 1999, pp 371-388.
[67] Baogang Hu,- George K. I. Mann, and Raymond G. Gosine, "New Methodology for Analytical and Optimal Design of Fuzzy PID Controllers », IEEE Transactions On Fuzzy Systems, VOL. 7, No. 5, October 1999, pp 521-539
[68] Zhi-Wei Woo, Hung-Yuan Chung, jin-Jye Lin, '' A PID type Fuzzy controller with self-tuning scaling factors'', Fuzzy sets and Systems , 2000, pp 321-326
[69] P.T.Chan, W.F.Xie, A.B.Rad, '' Tuning of Fuzzy controller for an open-loop unstable system: a genetic approach '', Fuzzy sets and Systems , 2000, pp 137-152
[70] K. S. Tang,- Kim Fung Man,- Guanrong Chen, Sam Kwong, "An Optimal Fuzzy PID Controller", IEEE Transactions On Industrial Electronics, Vol. 48, No. 4, August 2001, pp 757-765.
[71] Bao-Gang Hu,- George K. I. Mann, and Raymond G. Gosine, "A Systematic Study of Fuzzy PID Controllers—Function-Based Evaluation Approach »,IEEE Transactions On Fuzzy Systems, vol. 9, No. 5, October 2001, , pp 669-712
[72] A. Balestrino, A. Landi, and L. Sani, "CUK Converter Global Control Via Fuzzy Logic and Scaling Factors » IEEE Transactions On Industry Applications, Vol. 38, No. 2, March/April 2002, pp 406-413.
[73] Michail Petrov, Ivan Genchev, and Albina Taneva, " Fuzzy PID control of nonlinear plants", 2002 first international IEEE Symposium "Intelligent systems", September 2002, pp 30-35.
[74] P.G.Escamilla-ambrosio, N. Mort, " A novel design and tuning procedure for PID type fuzzy logic controllers", 2002 First International IEEE Symposium "Intelligent Systems", September 2002, pp 36-41.
[75] Ning wang, " A Fuzzy PID Controller For Multi-Model Plants", proceedings of the first international conference on machine learning and cybernetics, bijing, 4-5 November 2002, , pp 1401-1404
[76] I.S. Akkizidisa,*, G.N. Robertsa, P. Ridaob, J. Batlleb , "Designing a Fuzzy-like PD controller for an underwater robot", Control Engineering Practice , pp 417-480, 2003.
[77] Yu Yongquan- Huang Ying- Zeng Bi, "The Dynamic Fuzzy Method to Tune the Weight Factors of Neural Fuzzy PID Controller", IEEE 2004, pp 2397-2402.
[78] Jingwei Xu- Xin Feng, « Design of Adaptive Fuzzy PID Tuner Using Optimization Method", Proceedings of the 5" World Congress on Intelligent Control and Automation, June 15-19, 2004. Hangzhou, PR. China, IEEE, pp 2454-2458.
[79] Huiwen Deng, Yi Wang, « An Adaptive Fuzzy Logic Controller with Self-tuning Scaling Factors Based on Neural Networks*, 2005 IEEE, pp 392-396.

[80] Guihua Han, Lihua Chen, Junpeng Shao, Zhibin Sun, "Study of Fuzzy PID Controller for Industrial Steam Turbine Governing System", Proceedings of ISCIT2005, 2005 IEEE, pp 1228-1232.

[81] Jian Pei^1, Li-Ming Zhao^1, De-Jun Wang^1, Liang Chu, " Fuzzy PID Control Of Traction System For Vehicles", Proceedings of the Fourth International Conference on Machine Learning and Cybernetics, Guangzhou, 18-21 August 2005, IEEE , pp 773-777.

[82] Han-Xiong Li, , Lei Zhang, Kai-Yuan Cai, Guanrong Chen,'' An Improved Robust Fuzzy-PID Controller With Optimal Fuzzy Reasoning'', IEEE Transactions on Systems, Man, and Cybernetics—Part b: Cybernetics, Vol. 35, No. 6, December 2005, pp 1283,1294

[83] Y. Koike and K. Doya, "Multiple state estimation reinforcement learning for driving model: Driver model of automobile," in Proc. IEEE Int. Conf. Systems, Man, and Cybernetics (SMC), Tokyo, Japan, 1999,vol. 5, pp. 504–509,

[84] I. H. Suh, J. H. Kim, and F. C.-H. Rhee, "Fuzzy Q-learning for autonomous robot systems," in Proc. Int. Conf. Neural Networks, Houston, TX, 1997, vol. 3, pp. 1738–1743

[85] I. H. Suh, J. H. Kim, and S. R. Oh, "Region-based Q-learning for intelligent robot systems," in IEEE Int. Symp. Computational Intelligence Robotics and Automation (CIRA), Monterey, CA, 1997, pp. 172–178.

[86] Renders JM, "Metaphores biologiques appliquées à la commande de processus, Thèse de doctorat, Université libre de Bruxelles, 1994