A Fuzzy Neural Network Approach to Autonomous Vehicle Navigation

D. Kontoravdis, A. Likas, K. Blekas and A. Stafylopatis
National Technical University of Athens
Department of Electrical and Computer Engineering
Computer Science Division
157 73 Zographou, Athens, Greece

ABSTRACT: The paper presents an application of the GARIC architecture to autonomous vehicle navigation. This method constitutes a fuzzy neural approach that maps fuzzy control rules into the architecture of a neural network and uses a reinforcement learning scheme employing an adaptive heuristic critic to adjust the parameters of the fuzzy variables. Since the original GARIC formulation assumes continuous-valued control variables, we have developed a modified scheme that can be applied to problems assuming discrete control variables (as is the case with our motion control problem). Experimental tests on difficult grounds comprising left and right turns justify the effectiveness of the proposed method, since the vehicle learns to navigate almost perfectly in only a few steps and exhibits very steep learning curves compared to the pure reinforcement case.

1 Introduction

Reinforcement learning has been successfully applied to autonomous vehicle navigation in unknown environments. Previous attempts [4, 2, 3] assumed no prior knowledge about the task and the system to be controlled. Therefore, based on the received scalar reinforcement values, the learning system has to discover the most appropriate actions corresponding to the state of the environment as it is perceived through a set of sensors with which the vehicle is equipped [2].

In the present work, we have considered a fuzzy-neural reinforcement learning approach to the motion control problem, according to which prior knowledge is embedded into the action network in the form of fuzzy rules whose parameters have to be tuned while the vehicle operates, using the reinforcement values received by the environment. One recently proposed architecture that is based on the mapping of a fuzzy expert system into the architecture of a feedforward neural network and the subsequent adjusting of its parameters using reinforcement learning, is the GARIC (Generalized Approximate Reasoning-based Intelligent Control) architecture [1]. One characteristic of this approach is that it assumes continuous-valued outputs and, therefore, it is not straightforward to apply it to control problems that are characterized by a discrete action space, as is the case with our motion control problem.

To achieve this, we have developed a modification to the original GARIC scheme. Our approach considers an Action Selection Network (ASN) having a number of discrete output units, instead of one continuous output unit as is the case with the original GARIC formulation. This modification was necessary since, for our vehicle, the action selection at each step is restricted to a finite number of possible moves [2]. Moreover, neither a Stochastic Action Modifier (SAM) nor a defuzzification process is used, since at each step the strength of each rule contributes to the probability of selecting the corresponding action. The action that is eventually performed is selected using the probabilities provided at the output units of the ASN. Using the received scalar reinforcement and the corresponding reinforcement prediction provided by the Action Evaluation Network (AEN) we assign credit or blame to the selected action that is backpropagated (as an error value) through the ASN to adjust the parameters of the membership functions. The same error measure is also used to adjust the weights of the AEN.

The organization of this paper is as follows. In the next section the principles of the GARIC architecture are briefly discussed, while in Section 3 we present the proposed modifications in order to deal with control problems assuming a discrete action space. Finally in Section 4, we provide experimental results from the application of the modified arhitecture to an autonomous vehicle navigation problem and particularly to the control of the motion of an autonomous vehicle so that it can move without collisions through paths comprising left and right turns.

2 The GARIC Architecture

The Generalized Approximate Reasoning-based Intelligent Control (GARIC) architecture [1] results from the combination of neural networks, fuzzy sytems and reinforcement learning. It consitutes a new method of learning and adjusting the parameters of fuzzy systems using reinforcement values received from the environment.

The system is composed of two networks : the Action Selection Network (ASN) and the Action Evaluation Network (AEN). The first network maps an input state vector into an action, while the second (AEN) provides an evaluation of the current state. There is also a Stochastic Action Modifier (SAM) which receives the recommended action and an internal reinforcement signal and produces a final action that is actually applied to the physical system.

The AEN network is the adaptive critic element of the system. It is a typical two-layer feedforward neural network that receives the state of the system and produces a prediction of future reinforcement for this state. Morover, using the failure signal, the network evaluates the action suggested by the ASN network and produces an internal reinforcement signal. In the learning phase this signal is backpropagated through the network updating the weights of its links.

The architecture of ASN is suitable for mapping the rules of a fuzzy expert system. As it is shown in Figure 1, five layers are needed to implement the fuzzy inference process, each of them performing one stage of this process. The nodes of the input layer correspond to the linguistic variables of interest. The first hidden layer stores the antecedent conditions of the fuzzy rules. The number of its nodes is equal to the number of possible values of the linguistic variables. The operation that is performed is

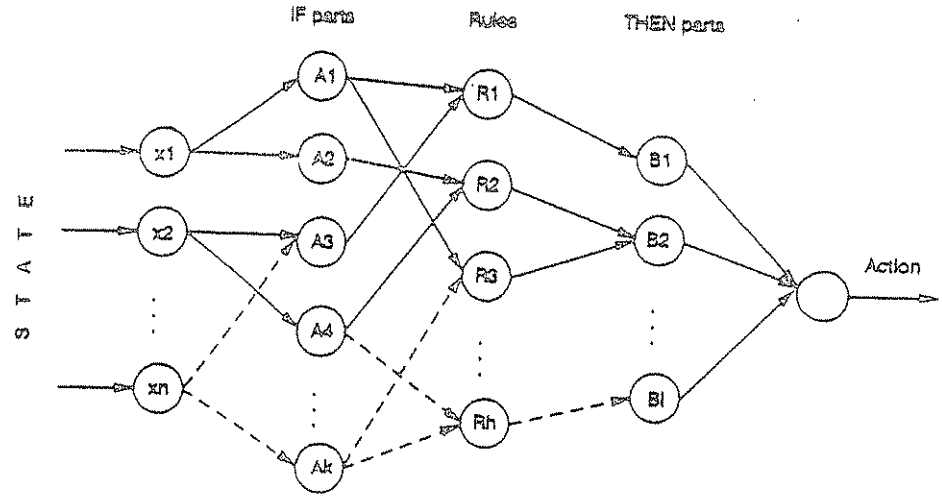

Figure 1: The architecture of the Action Selection Network

the fuzzification, i.e., the computation of the membership function values. Triangular shapes for the computation of membership values are preferable as they are simpler and more efficient.

The fuzzy rule bank is stored in the second hidden layer, thus, the number of nodes is equal to the number of fuzzy rules. The softmin operation takes place at each node providing the degree of applicability of each rule. The nodes in the third hidden layer correspond to the consequent parts of the fuzzy rules. The process of defuzzification is performed using the LMOM (local mean-of-maximum) method [1]. Finally the output layer contains as many nodes as is the number of the control variables. Each output node is connected to the nodes of the second and the third hidden layer and computes a *continuous* ouput value which corresponds to the action selected by the ASN network. During the learning phase the fuzzy control parameters of the network are updated in a reward/punishement fashion.

Finally, the SAM uses the internal reinforcement (provided by the AEN) and the recommended action F (suggested by the ASN) to stochastically generate an action F' which is a Gaussian random variable with mean F. This stochastic perturbation results in better exploration of the state and increased generalization ability.

3 The Proposed Approach

In its previously described original formulation the GARIC architecture assumes that the outputs of the ASN network take continuous values. In many cases there is the limitation that the control action assumes values from a discrete set of possible actions. One such case concerns the autonomous vehicle navigation application that is described in the next section. In order to deal with the requirement of discrete output space, we have developed a modification of the original GARIC architecture that mainly concerns the manner in which the defuzzification process is performed. The proposed architecture conciders an action selection network (ASN) having a number of discrete output units, instead of one continuous output unit as is in the case with the original

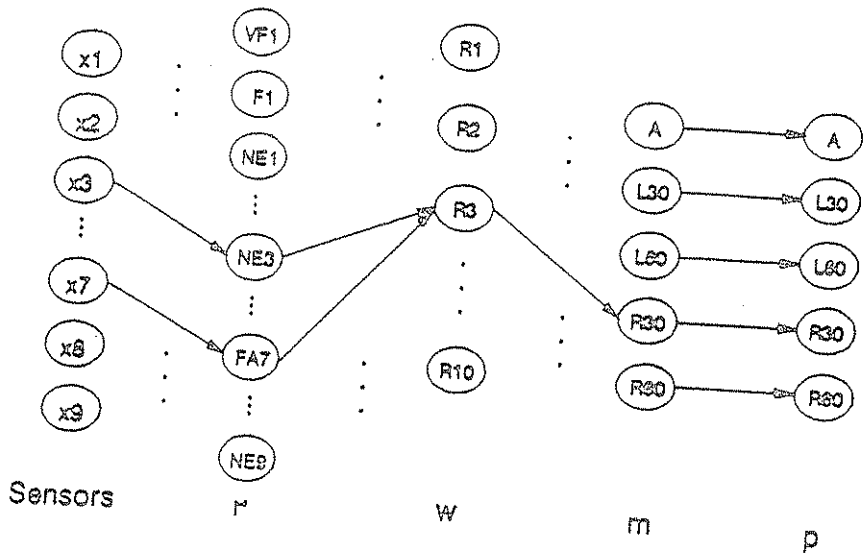

Figure 2: The modified Action Selection Network

GARIC formulation. At each step only one of the output units can be in the 'on' state, and the corresponding action is applied to the system. Furthermore, we are not using a stochastic action modifier since at each step the strength of each fuzzy rule contributes to the probability of selecting the corresponding action.

The AEN neural network is a two layer feedforward network that produces a reinforcement prediction $e(x)$ for a given input state x. The internal reinforcement r' that evaluates the action recommended by the ASN, is computed based on the reinforcement signal r and the prediction $e(x)$:

$$r' = r - e(x) \tag{1}$$

Equation (1) is different from the one used in [1], because we have to deal with an *immediate* reinforcement problem and not with a delayed one as is the case in [1]. The above internal reinforcement value plays the role of the error which is backpropagated to the network in order for the weights to be updated. Moreover, as we will see next, it is also used in the learning phase of the action selection network.

3.1 The Modified ASN

The ASN implements an inference scheme based on fuzzy control rules by providing at each step the control action that corresponds to the state of the system. It consists of five layers, as in the original GARIC formulation, but some of them operate in a different way.

Figure 2 displays the proposed architecture for the ASN. The first layer is the input layer consisting of the real-valued input vector that constitutes the state of the system. Each second layer unit corresponds to a possible linguistic value of an input variable and computes the triangular-shaped membership function value μ according

to the following equation:

$$\mu(x) = \begin{cases} 1 - \frac{|x-c|}{s_R} & \text{if } x \in [c, c+s_R] \\ 1 - \frac{|x-c|}{s_L} & \text{if } x \in [c-s_L, c) \\ 0 & \text{otherwise} \end{cases} \quad (2)$$

where c, s_L and s_R denote the centers, left spreads and right spreads of the linguistic variables.

Each node in the third layer corresponds to a rule of the fuzzy rule bank and receives the membership degrees of the values of the linguistic variables appearing in the if part of the rule. The output w_r of a node r provides the strength of the corresponding rule and is computed through the softmin operation:

$$w_r = \frac{\sum_i \mu_i e^{-k\mu_i}}{\sum_i e^{-k\mu_i}} \quad (3)$$

where the index i concerns all the nodes of the second layer that are connected to r.

The nodes in the fourth layer correspond to the possible output actions with inputs coming from all the rules which suggest the particular action. The output of each node i of this layer is computed as follows:

$$m_i = \frac{2}{1 + e^{-\sum_r (w_r - 0.5)}} - 1 \quad (4)$$

The above operation provides a way for computing the potential of action i based on the contributions w_r of the rules suggesting that output. Obviously, the potential m_i takes values in the range $[-1, 1]$.

The last layer (five) will have as many units as there are in the fourth layer (equal to the number of output actions). Each node i in this layer receives the potential m_i from the previous layer and, using the Boltzmann distribution, computes the normalized probability that the corresponding action is selected:

$$p_i = \frac{e^{m_i/T}}{\sum_j e^{m_j/T}} \quad (5)$$

where T is the temperature of the system. Random selection using this probability vector provides the control action α that is applied to the system.

Initially, the value of the parameter T is large, having all the output actions with almost the same selection probability (when $T \to \infty$, $e^{m_i/T} \to 1$ independently of the action i). Thus, in the first steps the stochastisity of the network is high investigating better every possible action. As learning proceeds, the temperature value gradually decreases, decreasing at the same time the stochasticity and biasing the probabilities towards selecting the action with the greatest potential.

3.2 Training of the ASN

The objective of learning in the ASN is the adjustment of the network parameters which are the centroid, left and right spreads corresponding to each linguistic value.

x	$\frac{\partial \mu}{\partial c}$	$\frac{\partial \mu}{\partial s_L}$	$\frac{\partial \mu}{\partial s_R}$
$[c, c+s_R]$	$1/s_R$	0	$(x-c)/s_R^2$
$[c-s_L, c]$	$-1/s_L$	$(c-x)/s_L^2$	0
otherwise	0	0	0

Table 1: The derivatives of the membership function with respect to its parameters

This can be achieved through rewarding a good selected action and blaming a bad one. In this sense, ASN backpropagates the internal reinforcement r' and the corresponding parameters are appropriately updated. This evaluation is used to update the centers and the spreads of the linguistic variables which are included in the fuzzy rules suggesting the selected output action α. Considering only the selected output α and traversing the network backwards, the learning equation for the parameter φ (where φ may be any of the c, s_R, s_L) can be written:

$$\begin{aligned} \Delta\varphi &= \eta r' \frac{\tilde{\partial} m_\alpha}{\partial \varphi} = \eta r' \sum_r \frac{\partial m_\alpha}{\partial w_r} \frac{\partial w_r}{\partial \varphi} \\ &= \eta r' \sum_r \frac{\partial m_\alpha}{\partial w_r} \frac{\partial w_r}{\partial \mu} \frac{\partial \mu}{\partial \varphi} \end{aligned} \tag{6}$$

with η being the learning rate of the ASN. Clearly, all the above derivatives can be computed locally at each node using the corresponding equations during the forward pass through the network.

From equation (4), it can be found that the derative $\partial m_\alpha / \partial w_r$ is computed as:

$$\frac{\partial m_\alpha}{\partial w_r} = \frac{1}{2}(1+m_\alpha)(1-m_\alpha) \tag{7}$$

where the index r concerns every rule suggesting output α. It is clear from the above equation that the derivative of m does not depend on w. Moreover, using equation (3), it can be shown that the partial derivative of w_r with respect to μ_j is the following:

$$\frac{\partial w_r}{\partial \mu_j} = \frac{e^{-k\mu_j}(1+k(w_r-\mu_j))}{\sum_i e^{-k\mu_i}} \tag{8}$$

Finally, Table 1 displays the derivatives of the membership values with respect to the centers and left and right spreads of the values of the linguistic variables used in the fuzzy rules. Using the derivatives provided from equations (7), (8) and Table 1, equation (6) can be used at each step to compute the nessecary updates of the network parameters.

4 Autonomous Vehicle Navigation

We have applied the modified GARIC architecture to an interesting control problem, concerning the collision-free autonomous navigation of a vehicle in various uknown

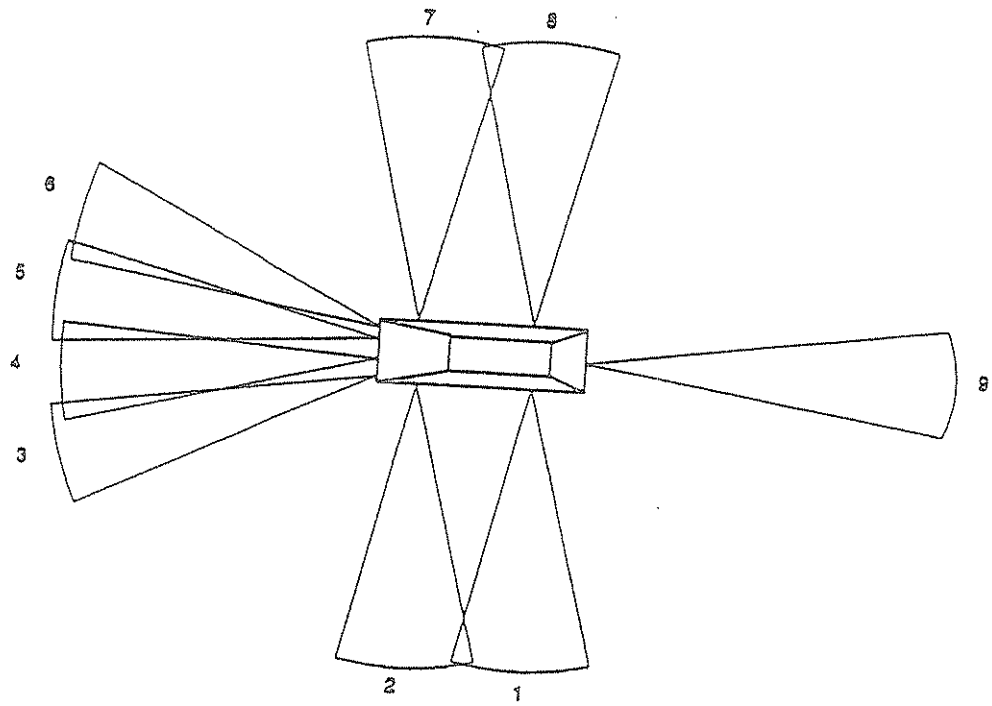

Figure 3: The positions of the sensors

grounds. We previously attempted to deal with this problem using pure connectionist reinforcement learning schemes assuming no a priori knowledge and our objective here is to investigate the benefits from using a fuzzy neural approach.

The experimental study of the problem has been performed through simulation using an appropriate graphical interface. The vehicle perceives its environment through the use of a number of sensors. The number and the configuration of the sensors play a very important role in the control task. On the one hand they cannot be too many as the purpose is to have a small knowledge of the environment, and on the other hand their number must be sufficient to provide the necessary information about the vehicle's environment.

Nine sensors seem to be adequate to describe the state of the vehicle at each time instant. As can be observed from Figure 3, four sensors are placed at the front, two at each side and one sensor at the back of the vehicle. Each sensor can detect the presence of an obstacle situated within a conic space in front of it and provides a measure of the distance of the obstacle. Specifically, the distance in the area tracked by each sensor is measured by a value in the range $[0, \ldots, 27]$ and the nine integer values provided by the sensors constitute the input state of the system.

The vehicle is able to perform one of five possible actions in response to the current state. These driving commands are:

A	:	Ahead
R30	:	30 degrees right
R60	:	60 degrees right
L30	:	30 degrees left
L60	:	60 degrees left

The evaluation of a state should be based on how probable it is for the vehicle to

Rule	*If*				*Then*
R1	VF3	VF4	VF5	VF6	A
R2	FA3	FA4	FA5	FA6	A
R3	NE3	FA7			R30
R4	VN3				R30
R5	VN3	VN4	VN5	FA7	R60
R6	VN3	VN4	VN5	VF7	R60
R7	FA2	NE6			L30
R8	VN6				L30
R9	FA2	VN4	VN5	VN6	L60
R10	VF2	VN4	VN5		L60

Table 2: The fuzzy rule bank

collide and go off the road. This probability can be estimated in terms of the positions of obstacles as they are detected by the sensors. The farther the obstacles are from the vehicle, the better the evaluation of the current state. The outcome of the evaluation is supplied to every unit of the neural network by means of a scalar reinforcement signal. In our system the reinforcement signal r is obtained as an average of the partial reinforcements r_i, $(i = 3, \ldots, 6)$, computed by the four front sensors according to the following rule:

$$r_i = \frac{\xi_i}{27} \tag{9}$$

where ξ_i take values from the set $\{3, \ldots, 27\}$. The values $\xi_i = 0, 1, 2$ indicate that the specific sensor i has detected an obstacle right in front of it, while larger values of ξ_i correspond to obstacles at a longer distance. If at least one of the sensors i satisfies the condition $\xi_i = 0, 1, 2$, the global reinforcement assumes a zero value.

In order to map the task of vehicle navigation into the action network, we have to introduce fuzzy linguistic variables together with fuzzy rules being able to describe the principles that the vehicle must respect for collision-free navigation. We have considered four linguistic variables, the same for each sensor :

VF	:	Very Far	[28...21]
FA	:	Far	[21...14]
NE	:	Near	[14...7]
VN	:	Very Near	[7...0]

The expressions next to the four linguistic variables describe the initial range of their values, which will be adjusted during the learning phase of the action network.

Ten rules are used to control the motion of the vehicle. These rules are shown in Table 2. The fist two rules are able to move the vehicle forward while the next four suggest right steering and the final four rules left steering. Each number following the linguistic variables in the fuzzy rules corresponds to one of the sensors. For example, the third rule (R3) means that if the third sensor detects a near obstacle, and the seventh detects a very far obstacle, then the vehicle must turn 30 degrees right.

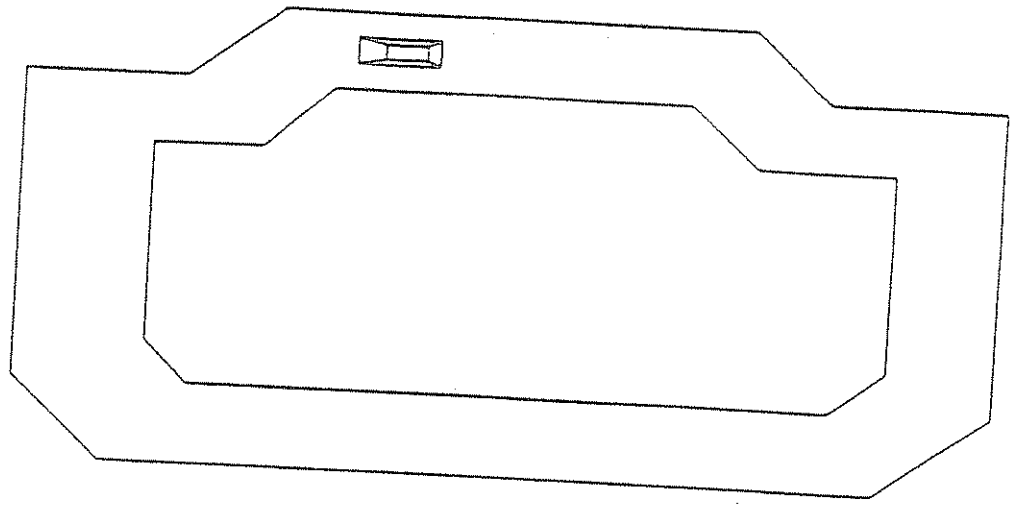

Figure 4: A typical ground

From the previous specifications, we can determine the size of each layer in the ASN. There are 9 inputs, 36 units in the second layer (equal to the number of linguistic variables), 10 units in the third layer (equal to the number of rules), 5 units in the fourth layer (equal to the number of possible actions) and, finally, 5 units in the last layer too. Figure 2 displays the architecure of this network, where the necessary connection weights corresponding to rule R3 are also depicted. In our experiments, the softmin parameter k was set equal to 0.1 and the value of the learning rate η was 0.05 for the ASN and 0.2 for the AEN. The initial value of T was 0.05 and if a cycle was achieved with length greater than 450 steps it was set equal to 0.005 Finally, the AEN network contained one hidden layer with 6 units and one output unit.

4.1 Experimental Results

Each experiment consisted of a sequence of cycles, where each cycle began with the vehicle at the same initial state and ended with a failure signal. At the start of each experiment the vehicle was placed at a randomly selected position. Statistical results of the effectiveness of learning during each experiment were obtained as follows. For smoothing purposes, at the end of each cycle an average value of the number of steps per cycle was computed by averaging over all cycles from the begining of the experiment up to that point. This representation aims at giving an overall view of the progress of learning without being affected by random fluctuations. Of course, under this style of presentation, the contribution of high scores is not readily visualized, since it slowly affects the average value.

To this end, a large number of grounds were tested using both the GARIC approach and pure reinforcement neural network techniques without fuzzy rules and assuming no a priori knowledge about the task [2, 3]. In the latter case, since the rules had to be discovered by the learning system, learning was very slow. In the initial stages, when the network was 'naive', the behavior of the vehicle was very unstable. The vehicle could not stay on the road before significant training was accomplished. However, at the end the vehicle exhibited good behavior, i.e. long cycles were accomplished, but after a large number of cycles. On the contrary, in the GARIC approach, due

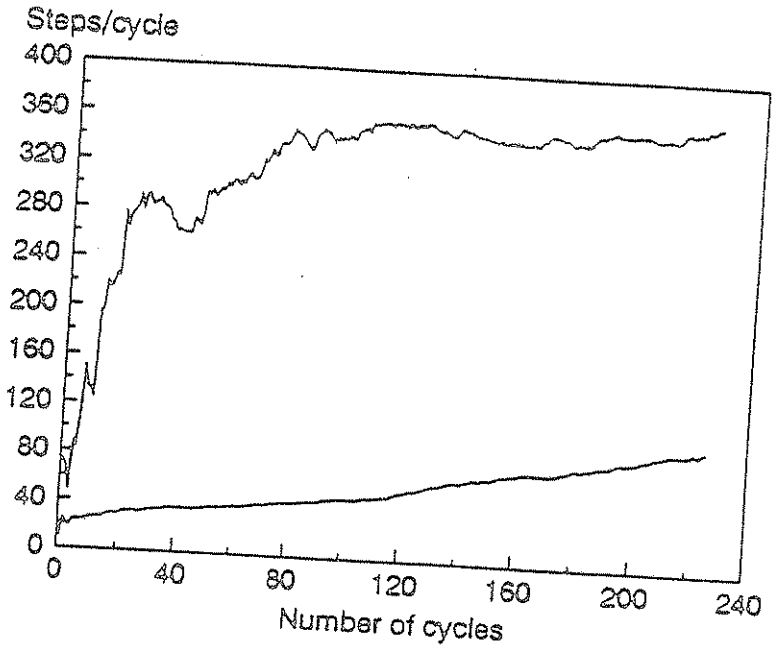

Figure 5: Performance of the vehicle using: (a) pure reinforcement learning and (b) the proposed GARIC approach

to the knowledge incorporated in the structure of the ASN, a small number of cycles was required to adjust the parameters of the fuzzy rules and reach a satisfactory performance level.

Among the grounds used in our experiments a typical one is shown in Figure 4. Figure 5 shows the performance of the system during learning to move on this ground, by presenting the average number of steps per cycle as a function of the number of cycles. As can be observed, the GARIC approach requires a small number of cycles to reach almost perfect (collision-free) behavior, while the performance of the pure reinforcement method is improving with a very small rate.

References

[1] H.R. Berenji and P. Khedkar, "Learning and Tuning Fuzzy Logic Controllers Through Reinforcements", IEEE Trans. on Neural Networks, vol. 3, No. 5, pp. 724–740, September 1992.

[2] D. Kontoravdis and A. Stafylopatis, "Reinforcement Learning Techniques for Autonomous Vehicle Control", Neural Network World, Vol. 3-4, pp. 329–346, 1992.

[3] D. Kontoravdis, A. Likas and A. Stafylopatis, "Collision-Free Movement of an Autonomous Vehicle Using Reinforcement Learning", Proc. European Conference on Artificial Intelligence (ECAI 92), pp. 666–670, Viena, August 1992.

[4] J. del R. Millan and C. Torras, "A Reinforcement Connectionist Approach to Robot Path Finding in Non-Maze-Like Environments", Machine Learning, vol. 8, pp. 363–395, 1992.