Evolutionary Fuzzy System for Architecture Control in a Constructive Neural Network

Rodrigo Calvo
Department of Computer Science and Statistics
University of São Paulo
Sao Carlos - SP, Brazil
rcalvo@icmc.usp.br

Mauricio Figueiredo
Department of Computer Science
State University of Maringa
87020-900, Maringa – PR, Brazil
mauricio@din.uem.br

Eric Aislan Antonelo
School of Information Science, Computer and Electrical Engineering
Halmstad University
Halmstad, Sweden
d404eran@stud.hh.se

Abstract – This work describes an evolutionary system to control the growth of a constructive neural network for autonomous navigation. A classifier system generates Takagi-Sugeno fuzzy rules and controls the architecture of a constructive neural network. The performance of the mobile robot guides the evolutionary learning mechanism. Experiments show the efficiency of the classifier fuzzy system for analyzing if it is worth inserting a new neuron into the architecture.

Index Terms - constructive neural networks, classifier systems, robot navigation.

I. INTRODUCTION

Autonomous navigations systems should be able to guide efficiently mobile robots to their goals without external assistance. Several works in the literature uses the computational intelligence approach to obtain robust and efficient systems [1, 2, 3]. In [4], innovative neuron models provide the navigation system the capacity for learning spatial concepts. Target seeking and obstacle avoidance behaviors are incrementally associated with specific features of objects (e.g. color) during the environmental interactions.

Constructive neural networks are systems that try to find automatically an optimal architecture for a specific problem starting from a minimal architecture [5, 6]. There are great prospects for applying constructive methods to the expansion control of neural networks architectures.

Evolutionary computational methods have been proposed as an efficient and robust alternative to the design of neural networks in this context [7, 8, 9].

This work describes an evolutionary system that supervises the expansion of a class of a constructive neural network described in [10]. Although simulation results show its good efficiency for acquiring navigation ability through a classical reinforcement learning strategy, eventually the architecture suffers from an excessive expansion. The proposed classifier system generates Takagi-Sugeno fuzzy rules for controlling the architecture of that constructive neural network. The performance of the mobile robot guides the evolutionary mechanism. Experiments confirm the effectiveness of the classifier fuzzy system for finding a good balance between the costs of the insertion a new neuron versus the gain of navigation performance.

The remainder of this work is organized as follows. Section 2 describes briefly the constructive neural network The proposed evolutionary fuzzy system is described in Section 3. Section 4 shows simulation results considering different experiments. A brief discussion about the results and future work possibilities are presented in Section 5.

II. CONSTRUCTIVE NEURAL NETWORK

The constructive neural network is designed for application in autonomous tasks, specifically in autonomous navigation systems [10]. A classical reinforcement learning strategy underlies the acquisition of the navigation strategy. Learning proceeds continuously from the start. Initially the neural network presents a poor performance, causing collisions against obstacles. Slowly the neural network acquires efficient and general navigation abilities. In the next it is presented a short description of the autonomous navigation system.

The navigation system consists of three main neural modules connected to an output neuron. Two of them, Obstacle Avoidance (OA) and Target Seeking (TS) modules, generate innate behaviors. A coordination module (CM) establishes (after learning) suitable weights for the behaviors generated by OA and TS modules. The weighted behaviors are combined in the output neuron (Fig. 1).

The OA and TS modules are neural networks with a priori knowledge that models their respective behavior (they do not learn). They are innate neural networks. Even if they operate independently the robot is still able to accomplish specific tasks.

The OA module generates the obstacle avoidance behavior. The inputs stem from the obstacle sensors and the outputs correspond to the adjustments for the steering angle. If only the OA module guides the robot, it does not collide. Unfortunately, it does not reach targets. The TS module generates the target seeking behavior. It receives inputs from the sensors that indicate direction of the target. If the TS module guides the robot, it is able to reach targets. However, if there is an obstacle between the robot and the target a collision occurs. As these modules are innate and aiming at keeping the analogy with biological systems, both neural networks are configured according to an evolutionary approach [11].

If the navigation system consists of both innate modules operating simultaneously and without coordination, there will be many conflicting situations and the navigation performance will certainly be poor. The function of the coordination module is to coordinate the instinctive behaviors generated by OA and TS innate modules, respectively.

The coordination module consists of three neuro-fuzzy networks: the Obstacle Distance (OD), Target Direction (TA) and Target Distance (TD) networks. They are connected to different sensorial fields: Obstacle Distance, Target Direction (angle) and Target Distance, respectively (Fig. 1). The architectures of the neural networks consist of two layers of fuzzy neurons [12]. The first layer is constructive (Fig. 2).

When a collision or a target capture events occur (at t_c or t_a iterations, respectively), the learning process is activated. Two main learning procedures succeed: a) weight adjustment and b) architecture expansion. Weight adjustments are based on the classical reinforcement learning strategy. After some learning events, the coordination module weights efficiently the innate modules, suitably balancing each behavior according to the actual environment situation.

The second learning procedure causes an insertion of a neuron into the architecture, specifically into the in the first layer of the networks (more details in [10]). For each instant t_c a neuron is inserted in the OD network. In a similar way, for each instant t_a a neuron is inserted in both TA and TD networks. Therefore, the architectures of the neural networks are changed always a learning interaction occurs. There are no criteria for evaluation of the expansion. Thus it is possible that eventually the architecture suffers from an excessive expansion.

As aforementioned, this work proposes an evolutionary system for controlling the undesirable growth of the OD, TA and TD networks. A classifier system generates takagi-sugeno fuzzy rules to decide when it is worth inserting a new neuron into the architecture. It is a difficult trade-off decision, since it implies on one side a cost for insertion a new neuron and on the other one a gain in navigation performance.

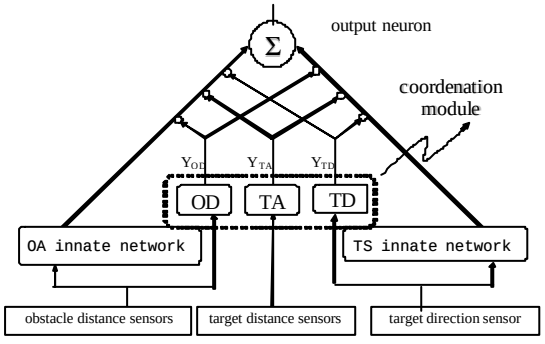


Fig 1. Autonomous navigation system.

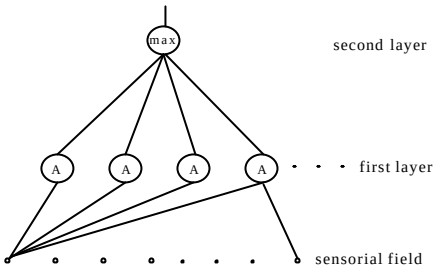


Fig 2. Architecture of constructive networks.

III. FUZZY CLASSIFIER SYSTEM

The Classifier System (CS) theory, paradigm proposed by Holland [13], is an evolutionary approach for generating adaptive inference mechanisms capable of operating in time-varying conditions. A CS is a mechanism for evolutive creation and update of knowledge represented by rules, called classifiers. They are <condition> - <action> rules with if-then mechanism. Differently from expert systems, a CS is a generic learning mechanism that can be used in several situations.

In the next it is described a hybrid evolutionary system, a fuzzy classifier system, designed to control the constructive neural architectures of the coordination module.

A. Architecture

This work uses a CS for defining fuzzy rules that monitors the network architectures of the coordination module described in Section II. Such classifier system interacts with the constructive neural networks by means of sensors and actuators and is composed by four main components: population of rules, credits setting module, reproduction module and competition module (Fig. 3). The sensors and actuators detect the current state of constructive neural networks and change its architecture, respectively (notice in Fig. 3 that the environment is the constructive neural network).

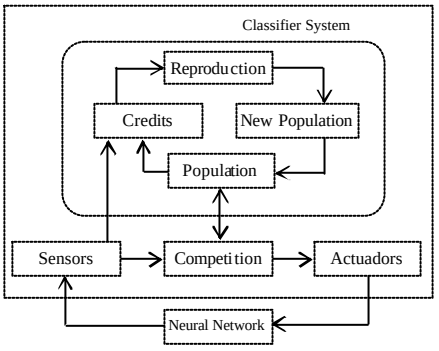


Fig 3. Architecture of Classifier System.

The population of fuzzy rules represents the knowledge of the system and it evolves during the robot navigation. The competition module receives the states of the OA, TA and TD networks (see Section II) and proceeds three inferences considering the rule population. Using the conclusions, the competition module decides if it is necessary or not the insertion of a new neuron.

The fuzzy classifier system interacts with the environment only at learning events (in the particular case of

this work, collision and target capture, at t_c or t_a iterations, respectively). At each target capture event, the performance of the coordination module (autonomous navigation system) is evaluated. If the navigation system performance decrease (considering the last two target capture events) then the population evolution is triggered.

Population of Rules: Each individual of the population is represented by <condition>-<action> rules, with if-then inference mechanism. Each rule can be described by a list of attributes, called chromosome. Each chromosome represents a fuzzy rule that composes the set of rules of the CS, and it is divided in three parts: fuzzy rule, fuzzy partition and consequent, as shown in Fig. 4.

The antecedent part of each fuzzy rule corresponds to the state of a specific network: amount of neurons in the neural network (*Amt*); growth rate of neural network (*Rate*); recognition efficiency of the input pattern (*Dist*); and frequency of low network output (under a level defined a priori) (*Freq*).

The universe of discourse for each input variable (related to each component that defines the state of a specific network) is divided in three linguistic values (Low, Medium and High). Such values are associated with membership functions, whose parameters are adjusted during the evolutionary process. The fuzzy system adopted is of type Takagi-Sugeno (the consequent part is a constant in [0,1]).

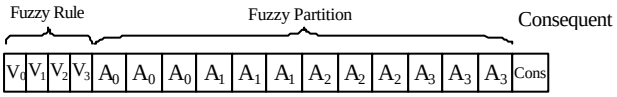


Fig 4. Structure of chromosome.

Thus the chromosome part related to the fuzzy rule is composed of four components, equivalent to the number of input variables of the system. Each component represents an input variable in the following sequence: *Amnt, Dist, Rate, Freq*. The number value of each component indicates the respective linguistic value. The association between the linguistic values and the component values is defined as: Low (0), Medium (1) and High (2). Fig. 5 shows two examples of individuals considering only the fuzzy rule chromosome part.

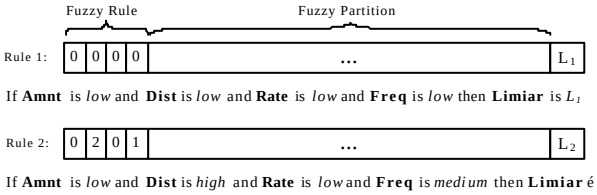


Fig 5. Examples of fuzzy rule.

In the chromosome fuzzy partition part, a linguistic value (general triangular membership function), respective to each input variable, is encoded in three components (*Dij, Lij, Rij*), where *i* indicates the linguistic value of the variable *j*, $0 = i = 2$ and $0 = j = 3$. The components are parameters of the membership function of the linguistic value indicated in the fuzzy rule part. They assume real non-negative values that are related to the reference position on the universe of discourse. Consider that TOP_{ij} is the element of universe of discourse such that the membership function degree is 1, where *j* and *i* define an input variable and a linguistic value, respectively. Then, *Dij* is the distance between the TOP_{ij} and $TOP_{(i-1)j}$ (if $i = 0$, then *D0j* is the distance between TOP_{0j} and the inferior limit of the universe of discourse); *Lij* and *Rij* are the distances between TOP_{ij} and the minimum and maximum elements of the support set, respectively (Fig. 6). Fig. 7 shows a chromosome for a general rule.

Initially, the rules are built so that all possible combinations among antecedents occur. As the fuzzy system is composed of four input variables and each variable consists of three linguistics values, the number of rules in the initial population is $3^4 = 81$. The consequents of the rules are randomly generated within the interval [0,1].

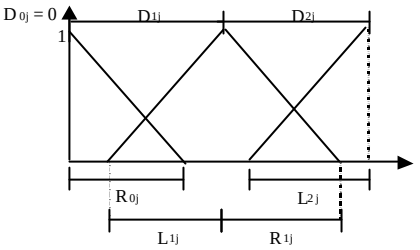


Fig. 6. Membership function encoding.

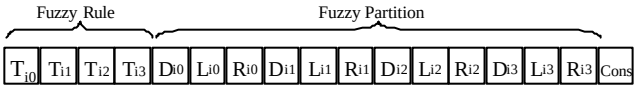


Fig. 7. Chromosome code for a general rule.

Competition Module: For each t_c iteration sensors detect the state of the OD and; at every t_a, iteration the states of TD and TA neural networks. The competition module finds the winner rule, that is, the rule whose antecedent is most similar to the state. There is a winner rule for related to each neural network (in the case of a t_a iteration). The similarity for each rule *r*, *S(r)*, is given by:

$$S(r) = \min_i(m_i(x)), \quad (1)$$

where: *x* is the respective state component.

The consequent of the winner rule is compared with the specific threshold associated with the respective neural network (OD, TA or TD). If the consequent is lower than the threshold then there is no expansion in the respective architecture. Otherwise, the consequent is compared with the inverse of the neural network output at the current learning event iteration. If it is higher, then a new neuron is inserted.

In any case, the threshold is adjusted according to Equation 2:

$$Threshold_{Net} = Threshold_{Net} + x(Cons - Threshold_{Net}), \quad (2)$$

where: $NET \in \{OA, TA, TD\}$, *Cons* is the consequent value of the winner rule and *x* is a constant within interval [0,1].

B. Evolution

The evolutionary process, responsible for evolving the population of rules, occurs at some target capture events. The evolutionary process takes place when it is verified a performance decreasing. The performance is defined according to the number of collisions that happens between two consecutive target captures. Thus, if the performance at a target capture iteration is better than the performance at the previous target capture iteration then a new population is generated.

Credit Setting Module: In this module, the rules receive credits according to their performance in the competition module. The value S(r) obtained by each rule through Equation 1 is sent to the credit setting module. The credit of the r^{th} rule is $C_r(t+1)$. After each generation, all credits are initialised, that is, $C_r(t)$=0. At each learning event, the credits are actualised according to Equation 3. Besides, this module also attributes credits to winner rules in the two last learning events occurrences. The attribution of credit to the winner rules in the last and the before the last learning events are given by Equations 4 and 5, respectively.

$$C_r(t+1) = C_r(t) + S(r) \quad (3)$$

$$C_r(t+1) = C_r(t) + zC_r(t) \quad (4)$$

$$C_r(t+1) = C_r(t) + dC_r(t) \quad (5)$$

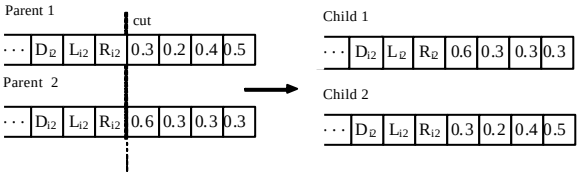
where: t is any moment t_c or t_a, indistinguishably, z and d are constants within the interval [0,1], such that $d \langle\langle z$.

Reproduction Module:

This module is responsible for initiating the process of generating a new population. For individual selection, it adopted the tournament selection method. [14]. Traditional evolutionary operators are applied to selected individuals.

The crossover operator takes a pair (parents) of selected individuals and, after executing the recombination in one point, two new individuals are generated. This operator is activated according to a specific probability. After applying the crossover operator, each individual may be chosen for mutation.

Fig. 8 shows an example of the action of a crossover operator. The second and third linguistic values of the fourth variable (*Freq*) are affected by the operator.



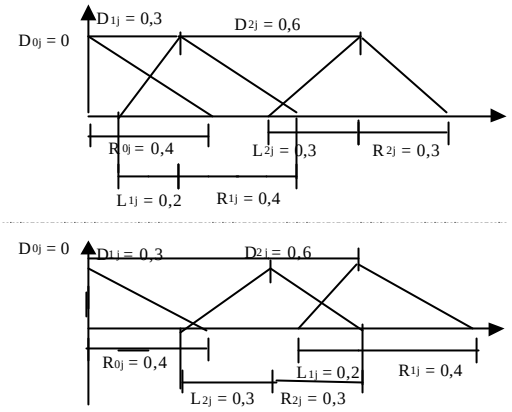


Fig. 8. Effect of crossover operator.

Fig. 9 shows two examples of the action of a mutation operator. The chromosomes that will suffer the action of the mutation operator are shown in Fig. 9(a). Only the three last components of the fuzzy partition part are considered (fourth fuzzy variable). This example shows the intervals (IM, IA, IB) where it is chosen values for the respective parameters (*Dij, Lij, Rij*) (Fig. 9(b)).

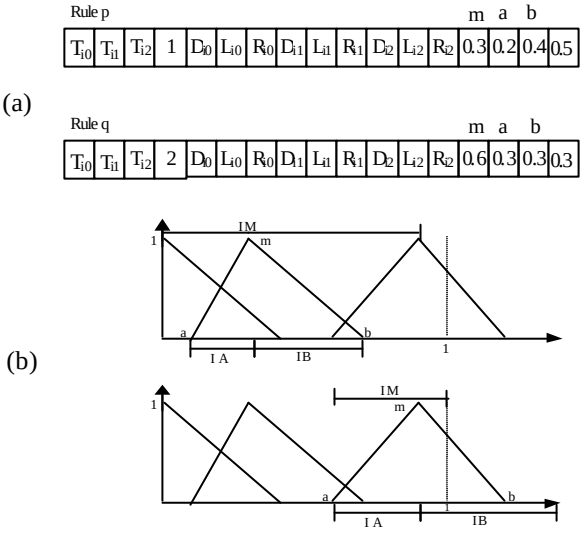


Fig. 9. Effect of mutation operator.

After the crossover and mutation operations, it is necessary to guarantee that every element of the universe of discourse presents a membership function degree different of zero for at least one of the linguistic values. Thus, it may be necessary to repair the membership functions. If repairing is necessary, the fuzzy membership function parameter (that the evolutionary operator has changed) is adjusted in a such way that the intersection between the support sets is equal to one discretization interval.

IV. Experiments and Results

This section shows experiments that confirm the capabilities of the fuzzy classifier system for architecture control. To do that the autonomous navigation system operates in two

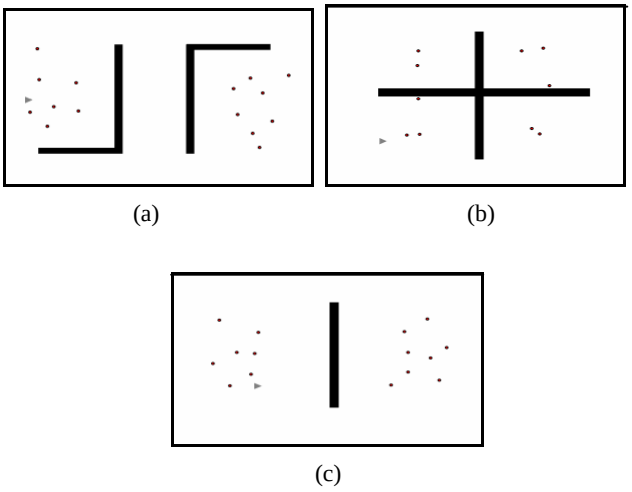
modes: firstly, the fuzzy classifier system is enable, and secondly, it is disable. Then the performance in these two cases are compared.

Fig. 10 (a), (b) and (c) shows the environments (1, 2 and 3, respectively) defined for evaluation. The rectangles represent obstacles, the cirles are target positions and the triangle is the robot. During simulation only one target is in the environment. After a capture, the target is eliminated and a new target is randomly positioned in the environment.

Other simulation parameters adopted for the fuzzy classifier systems are:

- $x = 0.5$, $z = 0.1$ and $d = 0.01$ (Equations 2, 4 and 5, respectively);
- $k = 2$ for the tournement selection method;
- mutation and crossover rates are 0.08 and 0.65, respectively;
- the threshold of each network is initialized to zero.

In the beginning, the threshold favors the insertion of neurons. As collision and captures occur, there is a trend for stabilization of the threshold.



(a) (b)

(c)

Fig. 10. Environments used for simulations: (a) Environment 1, (b) Environment 2; (c) Environment 3.

In the first experiment the fuzzy classifier system is disable. In this case, at each collision or capture learning events, a new neuron is always inserted in the OD network or in the TA and TD networks, respectively. This can be observed in Table 1. A total number of 15659 iterations are necessary for the robot complete its tasks in the three environments.

TABLE I
NUMBER OF NEURONS.

Environments	Collisions	Captures	OD	TD	TA
1	13	15	13	15	15
2	6	10	19	25	25
3	0	15	19	40	25

The graphic in Fig. 11(a) shows the changing of the number of neurons (only for the OD neural network) according to the number of collisions. Notice that they are linearly proportional (in an analogue way, the number of neurons increases linearly according to the number or target captures, for both TD and TA networks). The knowledge acquired while the robot is in the environment 1, is equally suitable for navigation in the other ones, that is, the navigation system learns a general navigation strategy (there is no collision for navigation in environment 3) Fig. 11(b).

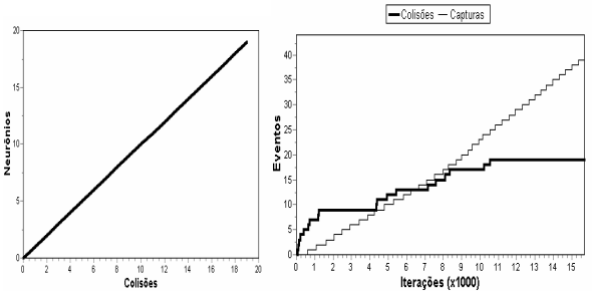


Fig. 11. (a) number of OD networks neurons x number of collisions; (b) learning events x number of iteractions.

The second experiment considers that the proposed fuzzy classifier system is controlling the expansion of the networks. The initial population is randomly generated (81 individuals).

Observe in Table 2 that the number of neurons is lesser than the number of collisions and captures. The simulation for the three environments required 18552 iterations and the evolutionary process is triggered 9 times. The values of threshold for OD, TD and TA networks, after stabilization, are 0.59, 0.609 and 0.609, respectively.

TABLE II
NUMBER OF NEURONS.

Environments	Collisions	Captures	OD	TD	TA
1	9	15	8	5	5
2	1	10	8	5	5
3	2	15	8	5	5

The graphic in Fig. 12(a) shows the relation between the number of collisions versus the number of OD network neurons. After the 8th collision, the architecture does not change any more. Similarly to the first experiment, the navigation system is capable to generalize the acquired knowledge and guide the robot in environments 2 and 3 (only 2 collisions occur during navigation in the environment 3) Fig. 12(b).

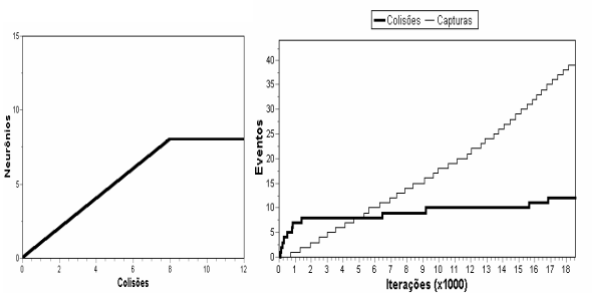


Fig. 12. (a) number of OD networks neurons x number of collisions; (b) learning events x number of iteractions.

Fig. 13 illustrates the configuration of the membership functions after the evolutionary process.

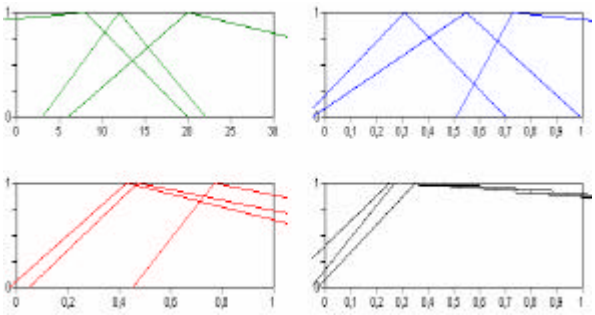


Fig 13. Linguistic membership functions for each of fuzzy variables: *Amnt*, *Dist*, *Rate*, *Freq* (from left to right and from top to down).

V. CONCLUSION

This work describes a fuzzy classifier system for the growth control of constructive network in autonomous navigation systems. The navigation system is based on modular hierarchical neural networks. There is a coordination network that learns a generalised navigation strategy via a reinforcement procedure. This network is of type constructive network, that is, for every learning event (collision or target capture) a neuron is always inserted into the architecture, if the fuzzy classifier system is not present [10].

The proposed fuzzy classifier system generates rules of Takagi-Sugeno type and controls the neural network architectures. For comparison purposes, two experiments are considered: E1 – the fuzzy classifier system controls the expansion architecture and; E2 – the fuzzy classifier system is not enable. Simulation results show an improvement performance if the fuzzy classifier system is enable (experiment E1): the number of neurons in the architecture is extremely low, if compared with the case of experiment E2. Although the number of iterations is smaller in the experiment E2 than in the experiment E1, the navigation system is able to capture all targets with good generalisation (it is observed that, in E1, the number of collisions that occur in the environment 1 is very higher than in the other different environments). Furthermore, the number of collisions in E2 is very higher than in the E1, that is, the fuzzy classifier system also improves the navigation system performance.

Future work includes: the extension of the fuzzy classifier system aiming at eliminating neurons from the network architectures (here the architecture only expands, but do not shrink); to establish mechanism to reduce membership function superposition and; to consider coevolutionary techniques.

REFERENCES

[1] Crestani, P. R., Figueiredo, M., e Von Zuben, F. (2002) "A hierarchical neuro-fuzzy approach to autonomous navigation", Proceedings of 2002 International Joint Conference on Neural Networks, (cd-rom), Honolulu, USA.

[2] Colombetti, M., Dorigo, M. and Borghi, G. (1996) "Behavior Analysis and Training – A Methodology for Behavior Engineering," IEEE Transactions on Systems, Man, and Cybernetics - Part B: Cybernetics, vol. 26, (3), pp. 365-380.

[3] Cazangi, R. e Figueiredo, M. (2002) "Simultaneous emergence of conflicting basic behaviors and their coordination in an evolutionary autonomous navigation system", Proceedings of 2002 IEEE Congress on Evolutionary Computation, (cd-rom), Honolulu, EUA.

[4] Antonelo, E. A., Figueiredo M., Baerveldt, A. and Calvo, C. (2005) "Intelligent Autonomous Navigation for Mobile Robots: Spatial Concept Acquisition and Object Discrimination". 6th IEEE International Symposium on Computational Intelligence in Robotics and Automation, Helsinki, Finland - unpublished.

[5] Eduardo, E. (2000) "Inteligência Computacional no Projeto Automático de Redes Neurais Híbridas e Redes Neurofuzzy Heterogêneas". Dissertação de Mestrado, UNICAMP, Campinas, SP.

[6] Fung, W. e Liu, Y. (2003) "Adaptive categorization of ART networks in robot behavior learning using game-theoretic formulation", Neural Networks, Vol. 16, no 10, pp. 1403 a 1420, dezembro.

[7] Chen, Z., Xiao, J., Cheng J. (1997) "PASS: A program for automatic structure search," in: Proceedings of the International Conference on Neural Networks, vol. 1, 1997, pp. 308-311;

[8] Moriarty, D. E. and Miikkulainen, R. (1998) "Hierachical evolution of neural networks," in: Proceedings of the Conference on Evolutionary Computation, pp. 428-433.

[9] Zhao, Q. (1997) "A co-evolutionary algorithm for neural network learning," in: Proceedings of the International Conference on Neural Networks, vol. 1, p. 432-437.

[10] Calvo, R. e Figueiredo M. (2003) "Reinforcement learning for hierarchical and modular neural network in autonomous robot navigation", Proceedings of 2003 International Joint Conference on Neural Networks – IJCNN, Oregon, USA.

[11] Figueiredo, M. and Gomide, F. (1996) "Evolving neurofuzzy networks for basic behaviors and a recategorization approach for their coordination," In: Genetic Algorithms and soft Computing (Herrera, F, and Verdegay, J. (Eds)), pp 533-552, Springer Verlag, USA.

[12] Gomide, F and Pedrycz W. (1998) "An Introduction to Fuzzy Sets: Analysis and Design". The MIT Press, Cambridge.

[13] Holland, J. H. (1975) "Adaptation in natural and artificial systems". The University of Michigan Press, Ann Arbor.

[14] Goldberg, D. E. (1989) Genetic Algorithms in Search, Optimization and Machine Learning, Addison-Wesley, Reading.