REACTIVE WITH TAGS CLASSIFIER SYSTEM APPLIED TO REAL ROBOT NAVIGATION

A. Sanchis*, J.M. Molina*, P. Isasi*, J. Segovia**

* Departamento de Informática, Universidad Carlos III de Madrid
Avda. Universidad 30, 28911- Leganés (Madrid)

** Departamento de Lenguajes y Sistemas, Facultad de Informática, UPM
Campus de Montegancedo, Boadilla del Monte (Madrid)

Abstract - *A Reactive with Tags Classifier System (RTCS) is a special Classifier System. This system combines the execution capabilities of symbolic systems and the learning capabilities of Genetic Algorithms. A RTCS is able to learn symbolic rules that allow to generate sequence of actions, chaining rules among diferent time instants, and react to new environmental situations, considering the last environmental situation to take a decision. The capacity of RTCS to learn good rules has been prove in robotics navigation problem. Results show the suitablity of this aproximation to the navigation problem and the coherence of extracted rules.*

1. INTRODUCTION

This work is centered in Classifier Systems, proposed by John Holland [6, 7, 8, 9, 10, 1, 4, 14]. A Classifier System is a kind of production system. In general, a production system is a set of rules that trigger others and accomplish certain actions [GD93]. Rules consist of a condition and an action. An action can activate the condition of other rule, and thus some rules interact on other. Classifier Systems are parallel production systems while traditional expert systems, generally, are not parallel. In a parallel production system several rules can be activated at the same time, while in not parallel ones, only one rule can be activated in each action. Together with the parallel activation capacity of rules, CS's have the property of learning rule chains sintactically simple to guide their behavior in changing environments, therefore they are considered as learning systems.

In traditional production systems, the value of a rule with respect to other is fixed by the programmer in conjunction with an expert or group of experts in the matter that is being emulated. In a CS does not exist this advantage. The relative value of the different rules is one of the key pieces of the information that it must be learnt. To facilitate this learning, the CS force rules to coexist in an information-based service economy. It is held a competition among rules, where the right to answer to the activation is going from the highest bidders, that will pay the value of their offers to those rules responsible of their activation. In this way, a middlemen chain is formed, that goes from manufacturers (the detectors) to consumers (actions toward the environment). The competitive nature of the economy assures that good rules (usefull ones) survive and bad rules disappear, as will be seen with more detail in the apportionment of credit algorithm.

A Classifier System consists of three principal components, that can be considered as activity levels. The first level (Action) is the responsible of giving answers (adequate or not) for the resolution of the outlined problem. In this level the rules of the system are found, codified in chains of characters over a restricted alphabet. The Action level produces a response to a given situation. The appropriateness of the given response to the problem to solve is measured through the reward that receives the rule from the environment. The second level (Credits Assignment) evaluates the results obtained in the previous level, distributing the rewards received by the rules that provide the output among rules that have contributed to the activation of each one of final rules (which give the output). As in other reinforcement learning methods, this evaluation can be adjusted applying a reward or payment from the environment, with a high value if the solution is profitable and a punishment or negative value if it is not. In this level, it does not possible to modify, however, the behavior of the system by means of changes in their rules, but it is possible to adjust their values and to establish, in certain measure, a good and wrong rules hierarchy. The task of the third level (Discovery) is to find new process that allows the system to discover new solutions. In this level a Genetic Algorithm is applied.

Although the search of new rules in CS is based on Genetic Algorithms application over a set of rules, a fundamental difference is the capacity of CS to generate isolated rules that are injected to a set of previously

0-7803-5670-5/99/$10.00 ©1999 IEEE

Authorized licensed use limited to: Univ Carlos III. Downloaded on March 26, 2009 at 11:24 from IEEE Xplore. Restrictions apply.

existing rules. Genetic Algorithms provide good results in many problems. However, as their analogous Genetic Programming and Evolutionary Strategies, the evaluation is accomplished on the complete system, without discriminating between different internal parts. If the system is composed of a set of rules, as in the case of the CS, an evaluation on the complete set without individualizing each one of the rules, does not permit to generate new isolated rules. Besides, the application of Genetic Algorithms over rules of Classifier Systems requires an intermediate representation, "codified rules", for genetic operators to act. Classifiers Systems are a specialized form of production system that have been designed to be amenable to the use of Genetic Algorithms [14].

CS are machine learning systems that learn syntactically simple string rules (called *classifiers*) to guide their performance in an arbitrary environment [10] and they work over tree fundamental concepts:

- The solution of the complete problem is a set of rules (a rules subset is a solution to concrete situations, even an isolate rule can be a solution for a specific situation, although this is not frequently). Relationship among rules is carried out in several internal cycles.
- Payment of each rule is distributed among rules that activated it in the internal cycles.
- Genetic Algorithm allows to generate rules from the better ones, producing, theoretically, an improvement in the global functioning of the system

The operation form of Classifier Systems presents some problems in execution time, in the learning of complex strategies, in the definition of the instant to call the GA and, as many other learning systems, in the presentation of the examples to the system. Centering in the first two problems, they are due to existence of internal cycles. These cycles permit the interrelationship among rules in order to produce elaborate solutions. While a CS executes internal cycles, remains isolated for the environmental information. This problem can be described as the necessity of a CS of being capable of "to react" to the stimuli of the environment. The attempts of seeking the "reactivity" in Classifier Systems have been approached from two different perspectives: the increase in the speed process of the system, with the systems ICS and hierarchic CS of Dorigo [3], and on the other hand, the execution of a rule for an input, without internal cycles and then without rules sequence, the HCA of Weib [20] based on Wilson [21] and Grefeenstete [5] works.

The problem of the capacity of these systems to learn rule chains appears because rule chains can not be broken between different learning instants. The loss of a rule of the chain could cause the loss of all the knowledge due to the interrelationships between rules. Isolated rules make no sense, but they are significant in groups, unknown a priori. These problems have been approached in the bibliography. Shu et al. [16], through the introduction of hierarchies in the CS. The hierarchy defines groups of rules that have been maintained along the learning process. These rules groups are formed a priori and they are built by an expert in the solution of the problem. The objective of these authors is to solve the same problem that DeJong [4] solved in Genetic Algorithms througth crowding.

In this work, a Reactive with Tags Classifier System (RTCS) has been designed, in the sense of Weib, but at the same time allows to elaborate complex strategies. For this, it is necessary to remember the definition of reactivity. A reactive system must decide for each input an action, and each action is determined by an input and in a CS, without losing the capacity of chaining rules in different time instants. For obtain a RTCS the operation of the action level has been modified. The solution proposed, therefore, must unite the capacity from learning without previous knowledge with the capacity of generating some kind of internal subdivision within CS to allow rule categories existence. To carry out this solution should be modified the codification of the rules (classifiers) and a field that represent the type or group which belongs each classifier has been included, named Tags.

2. REACTIVE WITH TAGS CLASSIFIER SYSTEMS

2.1 Reactive with Tags Classifier System Architecture, RTCS

A schematic representation of a traditional Classifier System is showed in figure 1. In these systems, it can be distinguished three activity levels:

(1) **Performance**, also called rule and message system: it interacts with the environment, gathering information through the input interface and producing the output through the output interface; it also receives the payoff. Structurally, the performance level consist of: (A) a finite population of fixed length condition/action rules, (B) a message list, (C) an input interface consisting of a set of environmental feature detectors and (D) an output interface for acting in the environment, that are also shown in figure 1.

(2) **Credit Assignment:** it causes rules to be established (fitting a rate of rules) on the basis of their observed utility to the systems goal.

(3) **Discovery:** it employs a Genetic Algorithm as a discovery operator that automatically generates new rules.

Authorized licensed use limited to: Univ Carlos III. Downloaded on March 26, 2009 at 11:24 from IEEE Xplore. Restrictions apply.

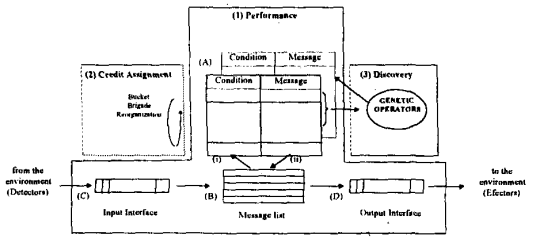

Figure 1: Representation of a Classifier System.

In a CS, rules are composed of two parts: condition and message and they are codified as strings: each condition is a string of fixed length k over the alphabet $\{0,1,\#\}$ (don't care symbols, "#", match both 0 as 1) and each message another string of fixed length k over the alphabet $\{0,1\}$.

Following this architecture, the performance level has been modified to learn reactions and actions. The performance level is composed by conditions and messages in the same way than a general CS except for two main differences: (1) condition/message length, k, is longer than the environmental message length, m, $(k>m)$, and (2) both conditions and messages are divided in three blocks. Each block contains different kind of information (Figure 2), environmental information, information related with rules fired in a previous instant (internal conditions) and information about the decisions.

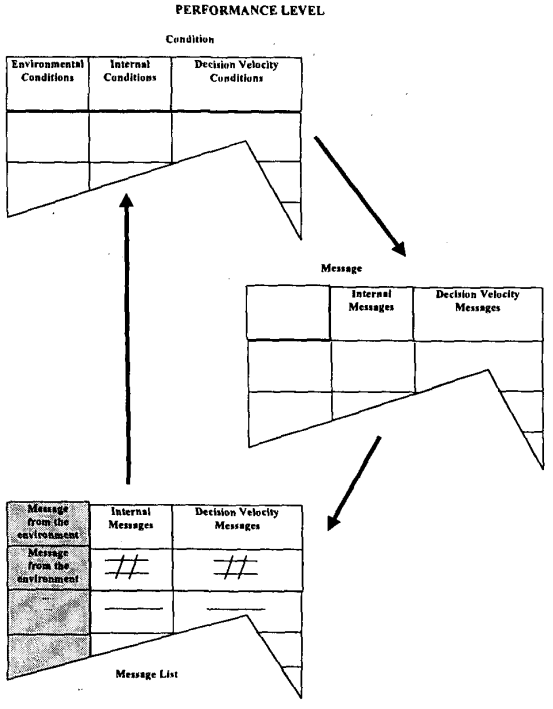

Figure 2: Performance level in the RTCS.

As it could be seen in Figure 2, the first block of the message, environmental block, is empty. This empty block is used to fuse the environmental message with messages of previous activated rules. The complete sequence of operations will be explained in more detail in section 2.2. This fusion mechanism allows the RTCS to learn complex actions, composed by a sequence of actions. Besides fused messages, another message with only the first block of message, environmental part, is posted to message list. This mechanism allows learning reactions, breaking the chain of rules.

2.2 Sequence of Operations in a Reactive with Tags Classifier System

In a traditional CS when a codified message arrives from the environment (through the input interface), the message is set in the message list. The message list is compared with all the classifiers and those that match with some message are fired. The fired rules post their messages into the message list. Several rules could be activated in parallel by a message. Before rules post messages, the message list ought to be cleaned. Activation of rules is repeated for n cycles. Finally, a message is chosen to give the output through the correspondent interface. The sequence of operations is summarized in Figure 3.

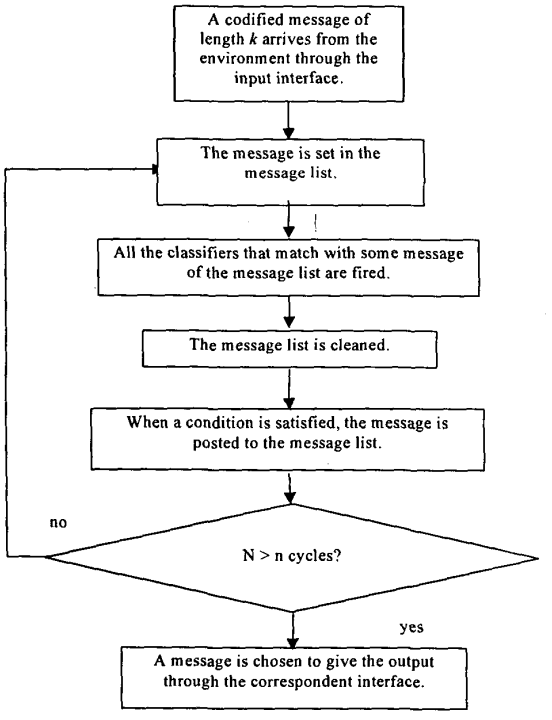

Figure 3: Representation of a Sequence of Operations in a Classifier System.

In a RTCS, when a codified message of length m arrives from the environment through the input interface, the message is fused with messages of previously activated

Authorized licensed use limited to: Univ Carlos III. Downloaded on March 26, 2009 at 11:24 from IEEE Xplore. Restrictions apply.

rules. A message composed with the environmental message and don't care symbols is posted to the message list. All the classifiers that match with some message of the message list are fired. A message is chosen from these fired rules. The list is kept to the next decision cycle. These operations do not contain the repetition of the matching process of the general CS, because the chain of rules needs the information of the next environmental input. The rule chain is over different inputs, using internal conditions and message fusion, allowing to learn reactions and actions sequence. The sequence of operations is summarized in Figure 4.

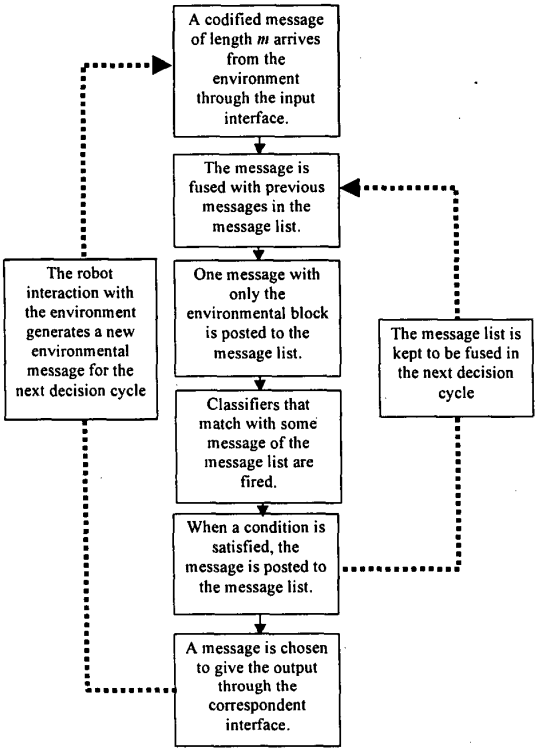

Figure 4: Representation of a Sequence of Operations in an Autonomous Robot Classifier System.

2.3 Knowledge Learned in RTCS

In this new SC, actions chaining is obtained taking into account two special mechanisms in conditions and messages: the environmental message is fused with the previously posted messages and internal conditions are added to evolve a chaining strategy. This strategy allows to chain rules activated by the environmental message with previous activated rules. In addition to the environmental message fusion, the CS requires the inclusion of internal conditions that provide the evolution of a chaining strategy. The fusion method gives the way to chain rules and the internal conditions support the knowledge about the relationship between rules. The evolution process over the internal conditions provided by the genetic algorithm leads to learn sequences of rules through time.

Although all the messages in the message list are composed by fusion, there is always one message with only the environment block filled with the environmental message (don't care symbols, "#", filling the other two blocks, see Figure 2). The matching process considers environmental conditions only and the system is able to break the rule chain and to react to the environment. In this way, reactions are obtained when a message, with the environmental information only, is posted to the message list.

These mechanisms allow the generation of more complex rules needed for the final solution of the problem. An example of Condition/Action rules that could evolve is as follows:

```
IF      External_Signal IS <type x>
        AND
        Last_Rule_Fired IS <type y>
        AND
        Decision_Velocity_Part IS <Vi, Vj>
THEN Send_Message <001...>
```

The reaction mechanism, on the other hand, allows the evolution of traditional reaction rules as:

```
IF      External_Signal IS <type x>
THEN Send_Message <001...>
```

3. APPLICATION OF RTCS TO THE NAVIGATION PROBLEM

Traditional Classifier Systems have demonstrated be appropriate learning systems. However, in many cases, their usefulness is reduced when the environment is dynamical, the problem is real and, furthermore, the behavior to learn is reactive. One of the environments where more clearly are found these characteristics is the robotic systems. The developed RTCS allows to overcome the limitations of traditional CS when are applied in these environments.

A fundamental requirement for autonomous robots is navigation, understanding navigation as a task that allows a robot to move, from place to place without danger neither damages. A classic example of architecture of control is the designated "subsumption", proposed by Brooks [2], that has been implemented with success in robots of MIT and other institutions. The subsumption architecture is based on behavior. Each behavior reacts in a situation and the global control is a behaviors composition. To implement these behaviors have been employed different systems: from finite state machines to fuzzy controllers. These behaviors rules could be designed by a human expert, designed "ad-hoc" for the problem, or learnt through some artificial intelligence techniques [12]. Some

Authorized licensed use limited to: Univ Carlos III. Downloaded on March 26, 2009 at 11:24 from IEEE Xplore. Restrictions apply.

approximations have employed Genetic Algorithms to evolve Fuzzy controllers [19], Evolutionary Strategies to evolve connections weights in a Braitenberg approximation [13], [11], or Neural Nets for behaviors learning [15].

When a CS is employed for learning reactive behaviors, an additional problem is detected respect to the action chains: these action chains blind the system, make it insensitive to the environment during the duration of the chain, since the system can not manage any new input during the decision process. If, furthermore, the environment where the learning is accomplished is dynamical, the system would have to read the sensors (input, situation of the environment) in each decision step, since this is the principal characteristic of reactive systems. For example, in the navigation of an autonomous robot through a dynamical environment problem studied (where the obstacles can be mobiles), robot would not have to remain blind any moment, therefore each movement must be the result of the application of a decision process over the last reading of the sensors. To solve this problem, a Reactive with Tags Classifier System, RTCS, is proposed [17], [18].

In the proposed learning system, the only previous system information is related to number of inputs (in the robot will be number of sensors), the domain, the number of outputs (in the robot, number of motors) and their description. Thus, the robot controller (the RTCS) starting without information about correct associations between sensors input and motors velocities. From this situation, the system (robot + controller) must be capable from learning to reach the greater degree of fitness to the sensors information.

The robot has to discover a set of effective rules, employing past situations experience, and must extract information of each situation, when this is produced. In this way, the system will learn from incremental way and the past experience remains implicitly represented through evolved rules.

3.1 Input and Output Codification

The codification of information in CS (the design of environmental and output messages) is based on the special problem where CS will be applied. In this work, the CS is used as a controller of an autonomous robot named Khepera [15]. The sensory inputs come in from eight infra-red proximity/ambient sensors. The robot has two wheels controlled by two independent DC motors with incremental encoder that allow any type of movement. Each wheel velocity could be read by a speedometer.

The sensors (proximity, ambient and speedometer) supply three kinds of incoming information: proximity to the obstacles, ambient light and velocity. Instead of using the eight infra-red sensors individually, they have been grouped giving a unique value from two sensor input values (Figure 6a), reducing the amount of information received by the CS. Representing the goal by a light source, the ambient information lets the robot know the angle (the angle position in the robot of the ambient sensor receiving more light) and the distance (the amount of light in the sensor) (Figure 6b).

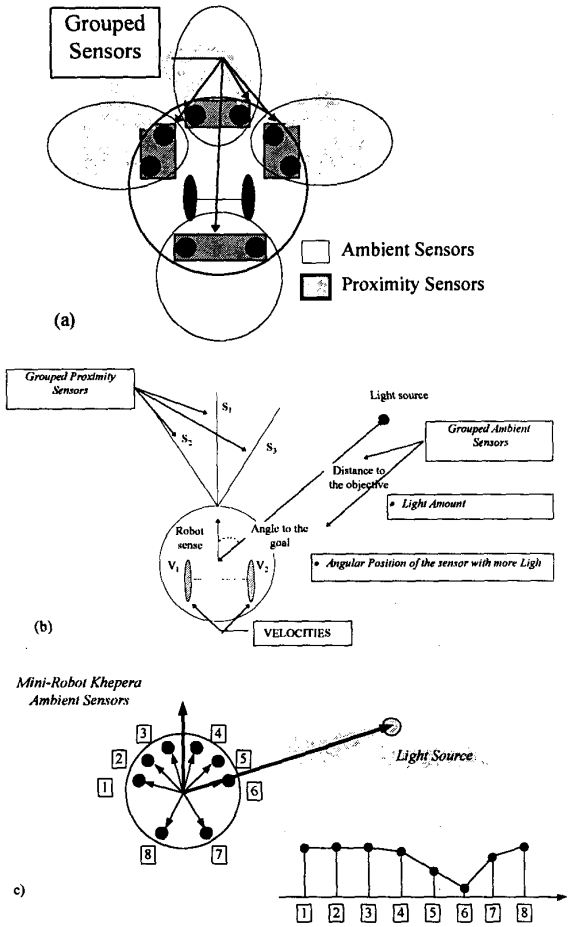

Figure 6: (a) Sensors considered in the real robot (b) Input information to the system (c)Light Incoming Distribution in the sensors.

Using the ambient sensors it is possible to measure the distance and the angle to a light source. The distribution of the amount of light coming into the eight sensors is used to evaluate the distance and the angle to the source. The amount of light received in the sensor depends on the distance of the light source. Each sensor is described by a sigmoidal function [15]. When the robot is placed near a light source, Figure 6(c), each sensor gives a value of light intensity based on the sigmoidal function. In Figure 6(c), an example of different values in each sensor is represented. In this case, the sensor 6 returns the minimum value from all sensors, the value is used to obtain the distance and the sensor number (ID 6) to obtain the angle to the light source.

Authorized licensed use limited to: Univ Carlos III. Downloaded on March 26, 2009 at 11:24 from IEEE Xplore. Restrictions apply.

The input to the CS consists of three proximity sensors, angle and goal distance (given by ambient sensors) and velocity values obtained by the speedometer. The outputs are the velocity values. The composition of the message could be seen in figure 7.

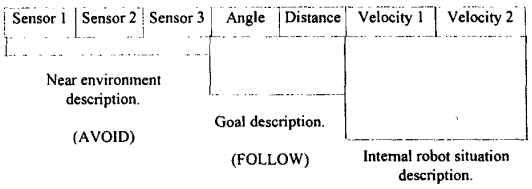

Figure 7: Composition of the environmental message.

The distance information of proximity sensors is obtained by the response curve of the sensors, that is a sigmoidal function defined over the intensity values domain. The distance domain is transformed, translating it into a simpler domain to codify the values. This transformation allows both the CS and the robot to be independent. So the CS could be developed for any robot by changing the transformation function. The input domain has been partitioned in four crisp sets. The maximum distance value "seen" by one sensor is 40 units and is divided in four equal sets. The angle sets are of different size to consider a fine fitting of the trajectory, avoiding big oscillations when the robot follows the right direction (the sets near 0 and 2π are smaller than the "$<\pi$" and the "$>\pi$" ones). To keep the independence between robot and CS, the distance values are translated from the real sensor values to a domain defined from 0 to ∞. The input domain has been partitioned in four crisp sets. Velocity values flow as input to the classifier system and as decision from the CS to the robot. The values are defined by the maximum and minimum velocities (10, -10). This range is divided in four equal sets. All these sets should be codified to build the message from the environment. Two binary digits are needed to represent each set. The codified inputs to the robot are displayed in the table 1.

Table 1: Input and Output Codification.

	Proximity	**Angle**
00	Very Near (VN)	Near 0 (0)
01	Near (N)	$<\pi$ (0-PI)
11	Far (F)	$>\pi$ (PI-2PI)
10	Very Far (VF)	Near 2π (2PI)

	Distance	**Velocities**
00	(0,25) (VN)	Slow Forward (F)
01	(25,100) (N)	Fast Forward (FF)
11	(100,200) (F)	Backward (Bc)
10	(200, ∞) (VF)	Stop (ST)

4. REAL ROBOT EXPERIMENTS

The accomplished experiments resemblance the accomplished in the simulator, in order to compare the results obtained by the same Classifier System both in the simulator and in a real environment, with the robot Khepera. The environment consists of several elements:

- A wood enclosure in white color, of 6.5 cm. of high, and 70 x 70 cm. of perimeter.
- A bulb of 2,5 V., placed in a foam chunk, and fed by a continuous current generator.
- The surface of the enclosure is covered of a black color cardboard, to asses the optimum behavior of the robot respecting to the source of light.
- Three objects have been placed on the enclosure in a similar way to the simulators world. These objects are white color, of 10 x 10 cm. and 6 cm. of height.

In Figure 8 a plan of the described real environment is shown. Three different starting positions of the Khepera appear. These positions are also similar to the ones used in the simulator.

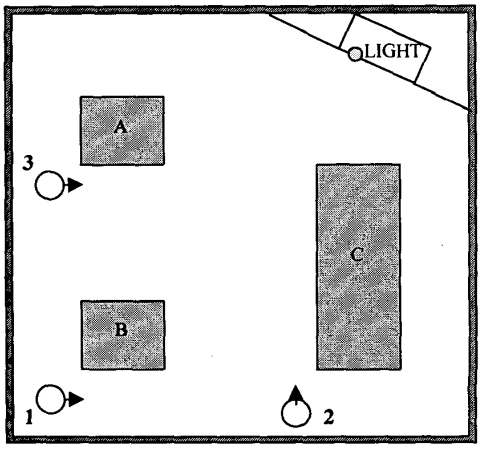

Figure 8: Schema of Real Robot Experiments.

12 experiments have been accomplished, consisting each of them of 20 consecutive robot executions. In the experiments, starting position and robot sense change (positions 1, 2 and 3), and also objects number objects A, B and C, with their possible combinations, eliminating an object). In each execution three objective parameters has been collected: number of produced collisions, time elapsed until arriving the objective, in seconds, and distance traveled, in centimeters. Furthermore, for each experiment, maximum and minimal values, average values and standard deviations have been calculated, for each one of these three parameters. In Figure 9 some comparative tables are shown.

Authorized licensed use limited to: Univ Carlos III. Downloaded on March 26, 2009 at 11:24 from IEEE Xplore. Restrictions apply.

	POS 1 ABC			POS 1 AB		
	Col	Time	Dis	Col	Time	Dis
Maximum	6	3:01	160,3	0	1:05	85,16
Minimal	0	0:55	67,13	0	0:48	64,74
Deviation	1,70	0:30	24,44	0,00	0:05	5,93
Average	0,60	1:18	83,51	0,00	0:54	70,04

	POS 1 AC			POS 1 BC		
	Col	Time	Dis	Col	Time	Dis
Maximum	4	1:44	119	4	2:36	179,9
Minimal	0	0:48	61,80	0	0:52	65,17
Deviation	0,91	0:17	18,74	0,99	0:26	29,51
Average	0,25	1:06	78,38	0,35	1:15	88,08

	POS 2 ABC			POS 2 AB		
	Col	Time	Dis	Col	Time	Dis
Maximum	2	1:55	129,8	0	0:40	49,79
Minimal	0	0:39	52,39	0	0:33	46,77
Deviation	0,45	0:20	18,93	0,00	0:02	0,95
Average	0,10	0:59	65,57	0,00	0:36	47,85

	POS 2 AC			POS 2 BC		
	Col	Time	Dis	Col	Time	Dis
Maximum	0	2:04	127,1	4	1:59	125,3
Minimal	0	0:42	53,31	0	0:44	53,89
Deviation	0,00	0:25	21,74	0,91	0:19	19,12
Average	0,00	1:04	69,50	0,25	1:02	71,78

	POS 3 ABC			POS 3 AB		
	Col	Time	Dis	Col	Time	Dis
Maximum	1	1:04	60,52	0	1:09	81,24
Minimal	0	0:32	43,50	0	0:34	43,46
Deviation	0,22	0:08	4,73	0,00	0:08	8,12
Average	0,05	0:42	47,28	0,00	0:39	48,74

	POS 3 AC			POS 3 BC		
	Col	Time	Dis	Col	Time	Dis
Maximum	2	1:14	72,57	0	0:38	45,52
Minimal	0	0:31	42,04	0	0:31	41,30
Deviation	0,49	0:11	7,88	0,00	0:02	1,33
Average	0,15	0:39	47,66	0,00	0:33	42,97

Figure 9: Results of RTCS in real robot.

As can be observed in these obtained data RTCS on a real robot operates almost without collisions in all situations and reached the goal in relatively short times (a similar duration). This results of the RTCS demonstrates that learned rules are useful for the navigation of a real robot with a stable and fixed functioning, so, behavior of the RTCS on the real robot demonstrates that degree of learning of the CS is sufficient to carry out the imposed task.

5. CONCLUSIONS

The main goal of this work has been a genetic learning system development, where the problem solution is defined through a set of rules, being each rule a part of the solution. These problems can be approached through Classifier Systems, but two disadvantages appear: execution time and complex strategies generation. These two problems are especially relevant in robotics, where rule systems are applied with relative efficiency and where, traditionally, they have been applied learning systems.

This goal has been approached from two different perspectives, on one hand, the need of finding a CS that could work with limitations in execution time and, on the other hand, the search of a solution to maintain the diversity in rules to elaborate complex strategies.

To provide a rapid response is a common requirement for any learning system, specially when works with temporary limitations. When Classifier Systems are applied in a changing environment, the temporary lag in taking decisions affects, on one hand, to the suitability of the output and, on the other hand to the output reward received from the environment. An unfitted reward produces a destructive effect on CS, since the reward is used to obtain the rules strength. The learning process is related with two levels (Credits Assignment and Discovery) that base their performance on the rule strength and such strength is the reward transposed to all rules through economy calculations among rules.

In order to develop a Classifier System able to learn reactions and complex strategies to survive in a dynamic world, the information, originating from the environment that CS receive, will be referred to previous CS situation. Therefore the environmental information is updated on the world state that surrounds the system in each instant and, it is injected in different instants at the same time that CS takes intermediate decisions. The output will not be only the action associated with the comparison of one rule, because solutions or intermediate decisions that contribute to the global solution are considered. In this way, an adequate sequence of these intermediate solutions considering actual environmental information gives the global solution.

This work has been centered in the development of a CS, named Reactive with Tags Classifier System (RTCS), that learns in a dynamic environment, with temporary restrictions in execution. The system contains a set of mechanisms that allow the incorporation of new environmental information in the process of taking decisions. This process allows rules sequence (chaining rules in different execution instants) and break the

Authorized licensed use limited to: Univ Carlos III. Downloaded on March 26, 2009 at 11:24 from IEEE Xplore. Restrictions apply.

sequence to provide a reactive output. Then, the developed RTCS has the capacity of learning reactions and strategies, so the dilemma between reactive and planned systems could be surpassed. A Classifier System that operates in this way allows to solve problems, independently of the environment in which are developed, without no previous knowledge of the problem, through the Credit Assignment and Genetic algorithms.

To evaluate the RTCS, a problem in a dynamic environment that required a reactive behavior has been elected. An environment with these characteristic is robotics, particularly in navigation of a robot, moving in a world with obstacles and where the robot goal is to reach a predefined point. This problem, from the point of view of learning, is considered sufficiently complex if the decision must be obtained in real time, since the environment continues changing during the time of taking a decision, or, from another point of view, the robot is moving while the decision is taken.

6. REFERENCES

[1] L. Booker, D.E. Goldberg y J.H. Holland, *"Classifier Systems and Genetic Algorithms"*, Artificial Intelligence, 235-282, (1989).

[2] R.A. Brooks, *"Intelligence Without Representation"*, Artificial Intelligence, 47, 139-159, (1991).

[3] M. Dorigo, *"ALECSYS and the AutonoMouse: Learning to Control a Real Robot by Distributed Classifier Systems"*, Machine Learning, 19, 209-240, (1995).

[4] D.E. Golberg, *"Genetic Algorithms in Search, Optimization, and Machine Learning"*. Addison Wesley, Reading MS, (1989).

[5] J.J. Grefenstette, *"Credit Assignment in Rule Discovery Systems Based on Genetic Algorithms"*, Machine Learning, vol 3, 225-245, (1988).

[6] J.Holland, *"Adaptation in Natural an Artificial Systems"*. University of Michigan Press, Ann Arbor, (1975).

[7] J.Holland, *"Adaptive Algorithms for Discovering and Using General Patterns in Growing Knoledge Bases"*, International Journal of Policy Analysis and Information Systems, vol 4, 245-268, (1980).

[8] J.Holland, *"Properties of the Bucket Brigade"*. In Proc. of International Conference on Genetic Algorithms and their Applications, vol 1, 1-7, (1985).

[9] J.Holland, *"A mathematical Framework for studying Learning in Classifier Systems"*, Physica D, 22, 307-317, (18986).

[10] Holland, J.H. *"Hidden order : how adaptation builds complexity"*. Reading (Massachusetts), Addison-Wesley, (1995).

[11] ST. Isasi, A. Berlanga, J. M. Molina, A. Sanchis, "Robot Controller against Environment, a Competitive Evolution", Special Session on Evolution Computation, 15th IMACS World Congress 1997 on Scientific Computation, Modelling and Applied Mathematics. Germany, (1997).

[12] V: Matellán, J.M. Molina, J. Sanz y C. Fernández, *"Learning Fuzzy Reactive Behaviors in Autonomous Robots"*, Procc. of the Fourth European Workshop on Learning Robots, Alemania, (1995).

[13] J.M. Molina, A. Sanchis, A. Berlanga, ST.Isasi-Viñuela, *"Evolving Connection Weight Between Sensors and Actuators in Robots"*. IEEE International Symposium on Industrial Electronics. (1997).

[14] M. Mitchell, *"An Introduction to Genetic Algorithms"*, MIT Press, Massachusetts, (1996).

[15] F. M Mondada y ST.I. Franzi , *"Mobile Robot Miniaturization: A Tool for Investigation in Control Algorithms"*. Proceedings of the Second International Conference on Fuzzy Systems. San Francisco, USA, (1993).

[16] L. Shu and J.Schaeffer, *"HCS: Adding Hierarchies to Classifier Systems"*, Proceedings of the 4th International Conference on Genetic Algorithms, 339-345, (1991).

[17] A. Sanchis, J.M. Molina y ST. Isasi, *"Classifier Systems for Learning Reactions in Robotic Systems"*, The first International Workshop on Machine Learning, Forecasting and Optimization (MALFO'96), 153-159 (1996).

[18] A Sanchis, J.M. Molina y ST. Isasi, *"Learning Reactive Behavior for Autonomous Robots using Classifier Systems"*, in Spatiotemporal Models in Biological and Artificial Systems, edited by F.L.Silva, J.C.Principe y L.B. Almeida. Vol. 37 of *Frontiers in Artificial Intelligence and Applications*, series editors: J.Breuker, R. López de Mántaras, S. Ohsuga y W. Swartout. Pag. 152-159, IOS Press,1997.

[19] M.A. Lee y H. Takagi, *"Integrating Design Stages on Fuzzy Systems Using Genetic Algorithms"*. Second International Conference on Fuzzy Systems, 612-617, (1993).

[20] G. Weiβ, *"Hierarchical Chunking in Classifier Systems"*, Proc. of the 12th. International Conference on Artificial Intelligence", 1335-1340, (1994).

[21] S. Wilson, *"Knowledge growth in an Artificial Animal"*, Proc. of the First International Conference on Genetic Algorithms and their Applications, 16-23, (1985).

Authorized licensed use limited to: Univ Carlos III. Downloaded on March 26, 2009 at 11:24 from IEEE Xplore. Restrictions apply.