# Learning Fuzzy Robot Controllers to Follow a Mobile Object

Manuel Mucientes
Dept. Electronics and Computer Science
University of Santiago de Compostela
Santiago de Compostela E-15782, SPAIN
Email: manuel@dec.usc.es

Jorge Casillas
Dept. Computer Science and Artificial Intelligence
University of Granada
Granada E-18071, SPAIN
Email: casillas@decsai.ugr.es

***Abstract*— The paper proposes a method to automatically design a fuzzy controller for the mobile object following behavior in mobile robotics. The system has been tested in several simulated situations using the Nomad 200 robot software. The proposed approach obtains a knowledge base with a good interpretability in a reduced time, and the designer only has to define the number of membership functions and the universe of discourse of each variable.**

## I. Introduction

Modern control architectures for mobile robots are hybrid, thus they are compound of layers [1]: at the lowest layer all the *reactive control* (selection of control action from the current sensorial information) is grouped, while on the top layer the *deliberative control* (planning tasks about future positions and actions of the robot) is done. In that way, the robot is able to implement complex tasks, and also react to changes in the environment (obstacles are moved, people appear, ...).

The reactive layer is usually implemented with behaviors (tasks like wall-following, go through a door, follow a person, avoid a moving obstacle, etc.) that are coordinated by the planning layer. The environments in which an autonomous robot moves are unconstrained, and have a high amount of uncertainty. Furthermore, information provided by robot sensors is noisy and unreliable. This problem becomes more important when using the ultrasound sensors data: low angular resolution and specular reflection. Fuzzy logic has shown to be a useful tool when dealing with this uncertainty and has been widely used for the design of behaviors in robotics [2].

The design of fuzzy systems requires a deep knowledge on the task to be controlled and forces to spend long time tuning the controller [3]. Due to that, in the last few years the use of learning methods for the design of fuzzy controllers has been generalized. There are different approaches: evolutionary algorithms [4]–[7], neural networks [8], [9], reinforcement learning [10]–[15], etc.

In this paper we present the automatic design of a fuzzy controller for the mobile object following behavior in mobile robotics using the COR (Cooperative Rules) methodology [16], [17]. To do that, we introduce a technique to automatically generate a training data set that represents state-action pairs for the whole input space in the mobile object following problem. The main advantages of the proposed approach (which combines the data set generation technique with the COR learning methodology) are the following:

- easiness in the design,
- very quick learning, and
- fuzzy controller with a good interpretability.

The paper is organized as follows. Section II introduces the mobile object following behavior. Section III presents the methodology that has been used. Section IV shows the obtained results and, finally, conclusions are discussed in Section V.

## II. Automatic Design of a Fuzzy Controller: Following a mobile object

In order to describe the methodology for obtaining a fuzzy controller in mobile robotics, the "follow a mobile object" behavior is going to be used as an example, but the same steps could be applied for learning other behaviors. Other results with the wall-following behavior are available in [18].

A mobile robot can implement the "follow a mobile object" behavior for tracking a person, or when it is cooperating with other robots in the implementation of a task and one of the robots is guiding the other ones. A good implementation of the behavior has to place the robot at the objective point $(x_{obj}, y_{obj})$ (Fig. 1). This point is defined using the desired distance ($d_{ref}$, Fig. 1) between the robot and the mobile object, and the reference deviation ($dev_{ref}$, Fig. 1), which is an angle that indicates the position of the robot with respect to the advance direction of the mobile object. If $dev_{ref} = 0$, the robot will follow the mobile object exactly behind it, while positive values of $dev_{ref}$ indicate that the robot will be placed at the right of the advance direction of the object, and negative values to the left. Also, a good controller for a behavior must implement smooth changes in velocity and angle of the robot.

Before explaining the learning methodology, some steps in the design of a behavior must be described.

### A. Preprocessing of the Variables

The first step in the design of the controller is the selection of the input and output variables. For this behavior, the input variables are:

- The distance between the robot and the objective point

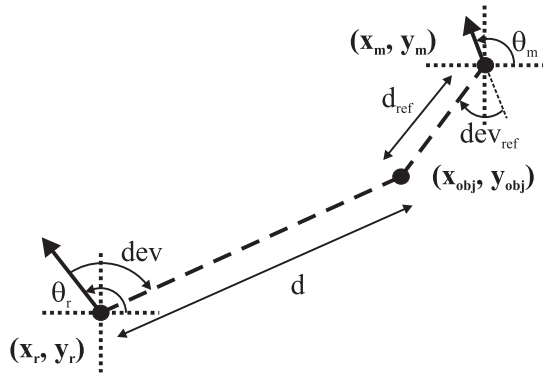



Fig. 1. Description of the points, distances and angles needed for the calculation of the input variables

$$d = \frac{\sqrt{(x_r - x_{obj})^2 + (y_r - y_{obj})^2}}{d_{ref}} \tag{1}$$

- The deviation of the robot with respect to the objective point. A negative value indicates that the robot is moving in a direction to the left of the objective point, while a positive value means that it is moving to the right.

$$dev = \arctan\left(\frac{y_{obj} - y_r}{x_{obj} - x_r}\right) - \theta_r \tag{2}$$

- The difference of velocity between the robot and the object

$$\triangle v = \frac{v_r - v_m}{maximum\, velocity} \tag{3}$$

- The difference of angle between the object and the robot

$$\triangle\theta = \theta_m - \theta_r \tag{4}$$

The output variables are the linear acceleration and the angular velocity.

### B. Universe of Discourse and Precision

The second step in the design is the definition of the universe of discourse, the number of fuzzy sets, and the precision ($p_n$) of each variable $n$. The universe of discourse is, for some variables ($d$, $\triangle v$, $\triangle\theta$), a reduced version of the real universe of discourse, and should contain those values of the variable that are meaningful for learning. For example, high values of distances are not useful during learning, because for all of them the robot will execute the same action. So, it is enough to include only a few high values in the universe of discourse.

The same occurs with the precision of the variables. Precision is used to generate the examples: very low values of $p_n$ will generate a higher number of examples, and many of them will not be meaningful because there will be very similar examples. Selecting valid values for the universes of discourse and the precisions is not difficult for somebody who has defined the input and output variables, and always it is possible to select an extended universe of discourse or a lower precision. In the worst case, a higher number of examples will be generated (some of them useless) and learning will take more time. For this behavior 6,435 examples have been used.

### C. Objective Function

In this methodology, it is of great importance the definition of the Scoring Function, $SF$ (equation 5), a function that scores the action of the rule base over an example. This function is behavior dependent, and for this behavior is defined as:

$$SF\left(RB\left(e^l\right)\right) = \alpha_1 + \alpha_2 + \alpha_3 \tag{5}$$

where $\alpha_1$, $\alpha_2$, and $\alpha_3$ are respectively:

$$\alpha_1 = 100 \cdot \frac{|d|}{p_d} \tag{6}$$

$$\alpha_2 = 10 \cdot \frac{|dev|}{p_{dev}} \tag{7}$$

$$\alpha_3 = \frac{|\triangle v|}{p_{\triangle v}} \tag{8}$$

$p_d$, $p_{dev}$, and $p_{\triangle v}$ are the precisions of the respective input variables. Precisions are used in these equations in order to evaluate the deviations of the values of the variables from the desired ones in a relative manner (the deviation of the value of variable $n$ from the desired one is measured in units of $p_n$). This makes possible the comparison of the deviations of different variables and, as a consequence, the assignment of the weights for each one of the variables. These weights (100, 10 and 1 for (6), (7), and (8) respectively) have been heuristically determined (no other values have been analyzed), and indicate how much important the deviation in the value of a variable is with respect to the deviation of other variables. The highest weight has been assigned to the distance, as the robot must be close to the objective point. An intermediate weight is associated to the deviation and, finally, the least important contribution to function $SF$ is for the difference in velocity.

The index that measures the global quality of the encoded rule set is:

$$f(RB) = \frac{1}{2 \cdot NE} \sum_{l=1}^{NE} \left(g(e^l)\right)^2 \tag{9}$$

where $NE$ is the number of examples, and $g(e^l)$ is defined as:

$$g(e^l) = \begin{cases} \left(1 - h(e^l)\right) \cdot \zeta + 1, & \text{if } h(e^l) \leq 1 \\ \exp\left(1 - h(e^l)\right), & \text{otherwise} \end{cases} \tag{10}$$

being $\zeta$ a scaling factor that has been set to 1000, and $h(e^l)$:

$$h(e^l) = \frac{\min\left(SF(e^l)\right) + 1}{SF\left(RB(e^l)\right) + 1} \tag{11}$$

where $\min\left(SF(e^l)\right)$ is the minimum score that an action can obtain for example $e^l$ (using only the discrete values of the output variables). These equations (9, 10, and 11) are independent of the behavior that is going to be learned.

### D. Robot Simulation

In order to reduce the time needed for learning, the simulation software of the Nomad 200 robot will only be used for testing the obtained controller. During learning, the movement of the robot will be modeled with the following set of equations (this model is valid but for all behaviors):

$$v_r^k = v_r^{k-1} + a^k \triangle t, \tag{12}$$

where $a^k$ is the linear acceleration at time $k$, and $\triangle t$ is the time between two control cycles (a value of $\triangle t = 1/3s$ has been used);

$$\theta_r^k = \theta_r^{k-1} - \omega^k \triangle t, \tag{13}$$

where $\omega^k$ is the angular velocity at time $k$; and

$$x_r^k = x_r^{k-1} + 2\, v_r^k \triangle t \, \cos\left(\frac{\pi}{2} - \theta_r^k\right), \tag{14}$$

$$y_r^k = y_r^{k-1} + 2\, v_r^k \triangle t \, \sin\left(\frac{\pi}{2} - \theta_r^k\right), \tag{15}$$

where $x_r^k$ and $y_r^k$ are the coordinates of the robot at time $k$.

The model assumes that the final $v_r$ and $\theta_r$ are reached without time delay. To simulate the inertia of the robot in its movements, the new position is calculated as if there were two control cycles between orders (2 in equations 14 and 15), so the selected accelerations and turnings are smoother (the robot will move a longer distance) and, on the contrary, decelerations must be harder.

### E. Construction of the Training Set

The controller learning is done using a set of examples. As has been mentioned, depending on the selected values for the universes of discourse and the precisions, the number of examples will be different. In this paper, 6,435 examples have been used. Its automatic generation is as follows: starting from the minimum value of each variable and increasing the value in a quantity equal to $p_n$ until the maximum value is reached, a number of different values for the variables is obtained. The set of examples is created combining these values for all the variables of the antecedent part.

The values of the variables of the consequent part for each example will be determined trying all the possible combinations of their discrete output values, and selecting those which let the robot reach the state closest to the ideal state (the state in which the robot is placed at the objective point and with a linear velocity that is equal to the velocity of the mobile object). The function that determines how much close a state is from the ideal state is $SF$: the lower the value of $SF$, the closer that the state is from the ideal state.

## III. Learning Methodology Based on COR

The process followed to learn the fuzzy controller is based on the COR methodology (proposed in [16] and extended in [17]). We have selected this process due to its good properties to quickly obtain knowledge bases with a high interpretability. The two following subsections describe the learning methodology and the proposed algorithm based on it.

### A. COR Methodology

A family of efficient and simple methods to derive fuzzy rules guided by covering criteria of the data in the example set, called *ad hoc data-driven methods*, has been proposed in the literature in the last few years. Their simplicity, in addition to their quickness and easy understanding, make them very suitable for learning tasks. However, ad hoc data-driven methods usually look for the fuzzy rules with the best individual performance (e.g. [19]) and therefore the global interaction among the rules of the rule base is not considered, thus involving knowledge bases with a bad accuracy.

With the aim of addressing these drawbacks keeping the interesting advantages of ad hoc data-driven methods, the COR methodology is proposed [16]. Instead of selecting the consequent with the highest performance in each subspace like these methods usually do, the COR methodology considers the possibility of using another consequent, different from the best one, when it allows the fuzzy system to be more accurate thanks to have a knowledge base with better cooperation.

COR consists of two stages:

1) *Search space construction* — It obtains a set of candidate consequents for each rule.
2) *Selection of the most cooperative fuzzy rule set* — It performs a combinatorial search among these sets looking for the combination of consequents with the best global accuracy.

A wider description of the COR-based rule generation process is shown in Fig. 2.

### B. COR Methodology with Ant Colony Optimization

Since the search space tackled in step 2. is usually large, it is necessary to use approximate search techniques. In [16], accurate linguistic models have been obtained using simulated annealing. However, since one of our constrains is to deal with a computational expensive evaluation function, in this paper the use of ant colony optimization (ACO) [20] is considered. It is a population search bio-inspired technique that considers heuristic information to allow it to get good solutions quickly. This section briefly describes the main components of the considered COR-based ACO algorithm, that was previously proposed in [21].

*1) Problem Representation for Learning Cooperative Fuzzy Rules:* To apply ACO in the COR methodology, it is convenient to see it as a combinatorial optimization problem with the capability of being represented on a weighted graph. In this way, we can face the problem considering a fixed number of subspaces and interpreting the learning process as the way of assigning consequents vectors—i.e., labels of the output fuzzy partitions—to these subspaces with respect to an optimality criterion (i.e., following the COR methodology).

Therefore, according to notation introduced in Fig. 2, each node $S_h \in S^+$ is assigned to each candidate consequent $(B_1^{k_h}, \ldots, B_m^{k_h}) \in C(S_h)$ and to the special symbol "don't care" ($R_\emptyset$) that stands for absence of rules in such a subspace. Fig. 3 shows the explored graph built from an example of

**Inputs**:

- *An input-output data set*—$E = \{e_1, \ldots, e_l, \ldots, e_N\}$, with $e_l = (x_1^l, \ldots, x_n^l, y_1^l, \ldots, y_m^l)$, $l \in \{1, \ldots, N\}$, $N$ being the data set size, and $n$ ($m$) being the number of input (output) variables—*representing the behavior of the problem being solved*.
- *A fuzzy partition of the variable spaces*. In our case, uniformly distributed fuzzy sets are regarded. Let $\mathcal{A}_i$ be the set of linguistic terms of the $i$-th input variable, with $i \in \{1, \ldots, n\}$, and $\mathcal{B}_j$ be the set of linguistic terms of the $j$-th output variable, with $j \in \{1, \ldots, m\}$, with $|\mathcal{A}_i|$ ($|\mathcal{B}_j|$) being the number of labels of the $i$-th ($j$-th) input (output) variable.

**Algorithm**:

1) *Search space construction*:

1.1. *Define the fuzzy input subspaces containing positive examples*: To do so, we should define the positive example set ($E^+(S_s)$) for each fuzzy input subspace $S_s = (A_1^s, \ldots, A_i^s, \ldots, A_n^s)$, with $A_i^s \in \mathcal{A}_i$ being a label, $s \in \{1, \ldots, N_S\}$, and $N_S = \prod_{i=1}^{n} |\mathcal{A}_i|$ being the number of fuzzy input subspaces. In this paper, we use the following:

$$E^+(S_s) = \{ \quad e_l \in E \mid \forall i \in \{1, \ldots, n\}, \; \forall A_i' \in \mathcal{A}_i, \mu_{A_i^s}(x_i^l) \geq \mu_{A_i'}(x_i^l) \; \} \tag{16}$$

with $\mu_{A_i^s}(\cdot)$ being the membership function associated with the label $A_i^s$.
Among all the $N_S$ possible fuzzy input subspaces, consider only those containing at least one positive example. To do so, the set of subspaces with positive examples is defined as $S^+ = \{S_h \mid E^+(S_h) \neq \emptyset\}$.

1.2. *Generate the set of candidate rules in each subspace with positive examples*: Firstly, the candidate consequent set associated with each subspace containing at least an example, $S_h \in S^+$, is defined. In this paper, we use the following:

$$C(S_h) = \{ \quad (B_1^{k_h}, \ldots, B_m^{k_h}) \in \mathcal{B}_1 \times \ldots \times \mathcal{B}_m \mid \exists e_l \in E^+(S_h) \; where \; \forall j \in \{1, \ldots, m\}, \; \forall B_j' \in \mathcal{B}_j, \; \mu_{B_j^{k_h}}(y_j^l) \geq \mu_{B_j'}(y_j^l) \; \} \tag{17}$$

Then, the candidate rule set for each subspace is defined as $CR(S_h) = \{R_{k_h} =$ [IF $X_1$ is $A_1^h$ and ... and $X_n$ is $A_n^h$ THEN $Y_1$ is $B_1^{k_h}$ and ... and $Y_m$ is $B_m^{k_h}$] such that $(B_1^{k_h}, \ldots, B_m^{k_h}) \in C(S_h)\}$.
To allow COR to reduce the initial number for fuzzy rules, the special element $R_\emptyset$ (which means "do not care") is added to each candidate rule set, i.e., $CR(S_h) = CR(S_h) \cup R_\emptyset$. If it is selected, no rules are used in the corresponding fuzzy input subspace.

2) *Selection of the most cooperative fuzzy rule set* — This stage is performed by running a combinatorial search algorithm to look for the combination $RB = \{R_1 \in CR(S_1), \ldots, R_h \in CR(S_h), \ldots, R_{|S^+|} \in CR(S_{|S^+|})\}$ with the best accuracy. Since the tackled search space is usually large, approximate search techniques should be used.
An index $f(RB)$ measuring the global quality of the encoded rule set is considered to evaluate the quality of each solution. In order to obtain solutions with a high interpretability, the original function is modified to penalize excessive number of rules:

$$f'(RB) = f(RB) + \beta \cdot f(RB_0) \cdot \frac{\#RB}{|S^+|} \tag{18}$$

with $\beta \in [0, 1]$ being a parameter defined by the designer to regulate the importance of the number of rules, $\#RB$ being the number of rules used in the evaluated solution (i.e., $|S^+| - |\{R_h \in RB \; such \; that \; R_h = R_\emptyset\}|$), and $RB_0$ being the initial rule base considered by the search algorithm.

Fig. 2. COR algorithm

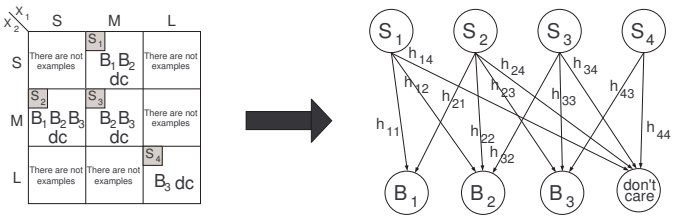



Fig. 3. Example of a graph built from sets of candidate rules generated by COR

candidate rule sets. To construct a complete solution, an ant iteratively goes over each rule and chooses a consequent with a probability that depends on the pheromone trail $\tau$ and the heuristic information $\eta$ associated to each decision. The order of selecting the rules is irrelevant.

*2) Heuristic Information:* The heuristic information on the potential preference of selecting a specific consequent vector, $B^{k_h}$, in each antecedent combination (subspace) is determined as described in Fig. 4.

For each subspace $S_h \in S^+$ do:

1) Build the sets $E^+(S_h)$ and $C(S_h)$ as shown in Fig. 2.
2) For each $B^{k_h} = (B_1^{k_h}, \ldots, B_m^{k_h}) \in C(S_h)$, make use of an initialization function based on a covering criterion to give a heuristic preference degree to each choice. In this paper, we use the following:

$$\eta_{hk_h} = \max_{e_l \in E^+(S_h)} Min\left(\mu_{A^h}(x^l), \mu_{B_j^{k_h}}(y^l)\right). \tag{19}$$

3) For each $B^{k_h} \notin C(S_h)$, make $\eta_{hk_h} = 0$.
4) Finally, for the "don't care" symbol, make the following:

$$\eta_{h,|\mathcal{B}_1|\cdot\ldots\cdot|\mathcal{B}_m|+1} = \frac{1}{\max\limits_{k_h \in \{1,\ldots,|C(S_h)|\}} \eta_{hk_h}}. \tag{20}$$

Fig. 4. Heuristic assignment process

*3) Pheromone Initialization:* The initial pheromone value of each assignment is obtained as follows:

$$\tau_0 = \frac{1}{|S^+|} \sum_{S_h \in S^+} \max_{B^{k_h} \in C(S_h)} \eta_{hk_h}. \tag{21}$$

In this way, the initial pheromone will be the mean value of the path constructed taking the best consequent in each rule according to the heuristic information (a greedy assignment).

*4) Fitness Function:* The fitness function will be the said objective function, defined in eq. (18) in Fig. 2.

*5) Ant Colony Optimization Scheme: Best-Worst Ant System Algorithm:* Once the previous components have been defined, an ACO algorithm has to be given to solve the problem. In this contribution, the BWAS algorithm [22] is considered. Its global scheme is shown in Fig. 5. The adaptation of these components to COR can be consulted in [21].

1) Give an initial pheromone value, $\tau_0$, to each edge.
2) While (*termination_condition* is not satisfied) do:
   a) Perform the track of each ant by the **solution construction process**.
   b) Apply the **pheromone evaporation mechanism**.
   c) Apply the **local search process** on the current-best solution.
   d) Update $S_{global\ best}$ and $S_{current\ worst}$.
   e) Apply the **Best-Worst pheromone trail update rule**.
   f) Apply the **pheromone trail mutation**.
   g) If (*stuck_condition* is satisfied) then apply **restart**.

Fig. 5. BWAS algorithm

## IV. Experimental Results

The learnt fuzzy controller has been tested using the Nomad 200 simulation software. The position, velocity and advance direction of the mobile object were directly obtained from the simulation software and passed to the control system in order to calculate the input variables. Tests have been carefully chosen, trying to present the controller with a wide range of situations of velocity and turning of the mobile object. It is important to remark that these tests have not been used during training. The training set is only composed of a list of examples (6,435) that have been chosen covering the input space with an adequate precision. These conditions warrantee that the quality of the learnt behavior does not depend on the movements of the mobile objects because the behavior can be generalized.

We have used the following parameter values for the COR-based ACO algorithm: 50 iterations, 30 ants, $\rho = 0.8$, $\alpha = 2$, $\beta = 2$, $P_m = 0.3$, $\sigma = 4$, $LSi = 10$, $LSn = 30$, and $R = 5$. No experiments were made with different values for these parameters. Therefore, the results shown below maybe could be improved with a more exhaustive parameter value selection. The learnt controller (shown in Fig. 6) has 112 linguistic rules and has been learned in only 36m (with an Intel(R) Pentium(R) III 1400 MHz processor) using a value of $\gamma = 0.2$ (eq. 18). If for any situation no rule is fired, then a null linear acceleration and angular velocity are selected. The maximum linear velocity the robot can reach is 61 cm/s, and the maximum angular velocity is $45^o/s$.

Figure 7 shows some trajectories of the robot when it is following different mobile objects at a reference distance of 1.5m and with a reference deviation of $0^o$. The trajectories are represented by circular marks. A higher concentration of marks indicates lower velocity. In order to visualize adequately both trajectories, in Fig. 7 the trajectory of the mobile object has been shifted in the y-axis direction. Thus, at the beginning (points $A_r$ for the robot and $A_m$ for the mobile object), both the robot and the object have the same $y$ coordinate, and their $x$ coordinate is the one represented in the figure (the robot is placed 1.5m to the left of the mobile object). The labels that have been placed along the trajectories represent a time gap between them of 3.3s (ten control cycles).

Ten tests have been done for each one of the three analyzed types of trajectories (Fig. 7). The average values measured for some parameters that reflect the controller performance are shown in Table I. These parameters are the average distance error ($\delta d = |d - d_{ref}|$), the average deviation error ($\delta dev = |dev - dev_{ref}|$), and the average velocity change. The latter parameter measures the change in the linear velocity between two consecutive cycles, reflecting the smoothness of the behavior (a low value indicates a smooth behavior).

TABLE I
Average values of some parameters for the three types of trajectories

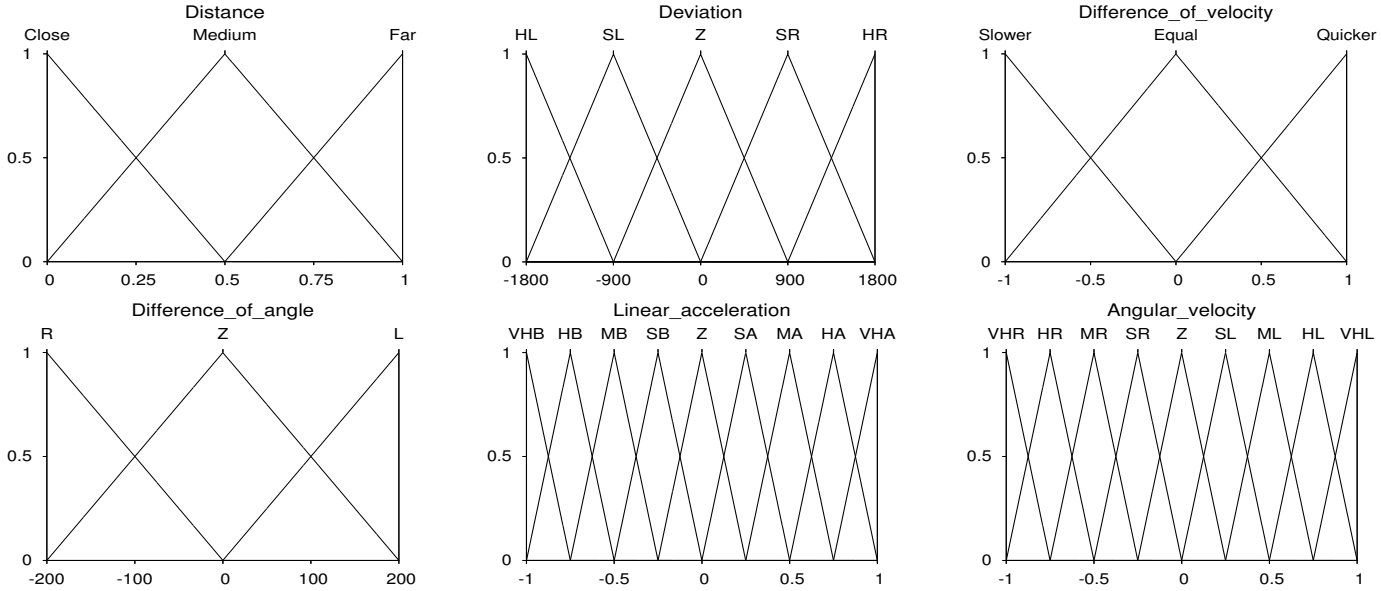
| | $\delta d$ (cm) | $\delta dev$ (degrees) | Vel. change (cm/s) |
|---|---|---|---|
| Fig. 7(a) | 29 | 22 | 6.13 |
| Fig. 7(b) | 10 | 4 | 11.71 |
| Fig. 7(c) | 20 | 10 | 6.47 |

In order to show the accuracy of the controller, the three trajectories of Fig. 7 are going to be described:

- Fig. 7(a): this example shows a behavior of the mobile object that makes quite difficult to implement the "follow a mobile object" task. At the beginning the robot is placed at $A_r$ and the object is placed at $A_m$ (remember that really the $y$ coordinate is the same for both points). The mobile object has a linear velocity of 38 cm/s along all the path, and implements turnings with the maximum angular velocity ($45^o/s$). These sudden and very sharp changes in direction make very difficult for the robot to be at the right reference distance and with the adequate reference deviation in the next control cycles. As a result, the errors are the highest ones of the three types of trajectories (Table I).
- Fig. 7(b): this is the easiest example, as there are few turns of the mobile object. The object moves with a high velocity (51 cm/s) except between points $B_m$-$C_m$ and $G_m$-$H_m$, where velocity is decreased to 25 cm/s in order to test the controller. Also, turning are implemented with an angular velocity of $30^o/s$. With these conditions, the errors in distance and deviation are low, but the change in velocity is high due to the abrupt changes in the speed of the object.
- Fig. 7(c): the last type of trajectory is also quite difficult because the mobile object is changing its movement direction for a long time. Between points $A_m$-$C_m$ and $G_m$-$L_m$ the object moves straight and at 38 cm/s, but between $C_m$-$G_m$ the speed is increased to 51 cm/s, and a continuous turning at $20^o/s$ is implemented. Due to this continuous change in the direction of the mobile object, the values of the errors (Table I) take a value higher than the previous type of trajectory.

As a resume, the accuracy of the controller is good, but when the mobile object implements continuous or sharp changes in direction the controller needs a few control cycles to reach the reference distance and deviation. On the other

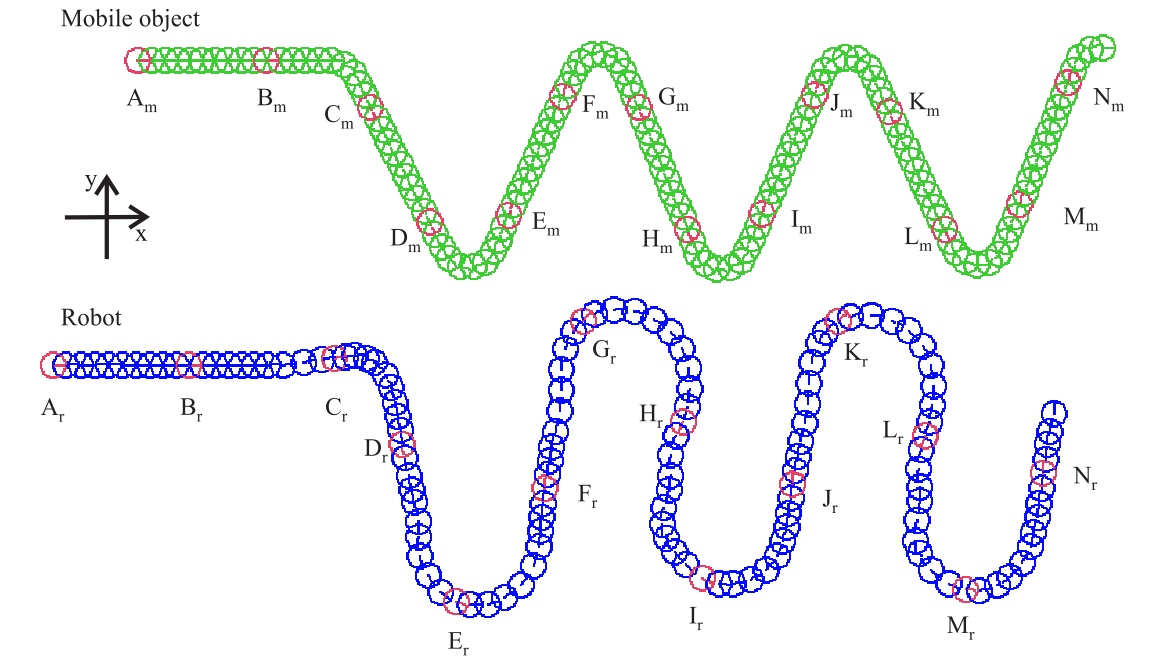
**(a)** Data base



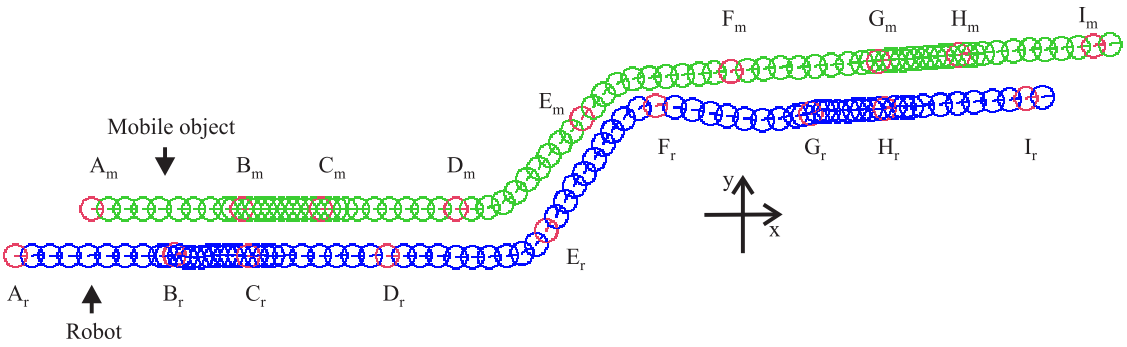

**(b)** Rule base

| *Rule* | *d* | *dev* | $\triangle v$ | $\triangle\theta$ | *Lin. acc.* | *Ang. vel.* | *Rule* | *d* | *dev* | $\triangle v$ | $\triangle\theta$ | *Lin. acc.* | *Ang. vel.* |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $R_1$ | Near | HL | Slower | R | HA | MR | $R_{57}$ | Medium | Z | Equal | R | SA | SL |
| $R_2$ | Near | HL | Slower | Z | SA | SL | $R_{58}$ | Medium | Z | Equal | Z | MA | SL |
| $R_3$ | Near | HL | Slower | L | HA | VHL | $R_{59}$ | Medium | Z | Equal | L | MA | HL |
| $R_4$ | Near | HL | Equal | R | Z | MR | $R_{60}$ | Medium | Z | Quicker | Z | SB | SL |
| $R_5$ | Near | HL | Quicker | R | VHB | MR | $R_{61}$ | Medium | SR | Slower | R | HA | VHL |
| $R_6$ | Near | HL | Quicker | Z | VHB | SL | $R_{62}$ | Medium | SR | Slower | Z | VHA | |
| $R_7$ | Near | HL | Quicker | L | HB | | $R_{63}$ | Medium | SR | Slower | L | VHA | |
| $R_8$ | Near | SL | Slower | Z | HA | HL | $R_{64}$ | Medium | SR | Equal | Z | SA | |
| $R_9$ | Near | SL | Slower | L | HA | VHL | $R_{65}$ | Medium | SR | Equal | L | MA | |
| $R_{10}$ | Near | SL | Equal | R | SB | HR | $R_{66}$ | Medium | SR | Quicker | R | HB | |
| $R_{11}$ | Near | SL | Equal | L | Z | HL | $R_{67}$ | Medium | SR | Quicker | Z | HB | |
| $R_{12}$ | Near | SL | Quicker | R | HB | MR | $R_{68}$ | Medium | SR | Quicker | L | MB | |
| $R_{13}$ | Near | SL | Quicker | Z | VHB | HL | $R_{69}$ | Medium | HR | Slower | R | Z | |
| $R_{14}$ | Near | SL | Quicker | L | VHB | VHL | $R_{70}$ | Medium | HR | Slower | Z | SA | |
| $R_{15}$ | Near | Z | Slower | R | HA | Z | $R_{71}$ | Medium | HR | Slower | L | SA | |
| $R_{16}$ | Near | Z | Slower | Z | HA | HL | $R_{72}$ | Medium | HR | Equal | R | SB | SL |
| $R_{17}$ | Near | Z | Slower | L | VHA | VHL | $R_{73}$ | Medium | HR | Equal | Z | HB | |
| $R_{18}$ | Near | Z | Equal | R | Z | MR | $R_{74}$ | Medium | HR | Equal | L | SB | |
| $R_{19}$ | Near | Z | Equal | Z | Z | SL | $R_{75}$ | Medium | HR | Quicker | R | VHB | SL |
| $R_{20}$ | Near | Z | Equal | L | Z | VHL | $R_{76}$ | Medium | HR | Quicker | L | VHB | |
| $R_{21}$ | Near | Z | Quicker | R | HB | MR | $R_{77}$ | Far | HL | Slower | R | SB | |
| $R_{22}$ | Near | Z | Quicker | Z | HB | SL | $R_{78}$ | Far | HL | Slower | Z | SB | SL |
| $R_{23}$ | Near | SR | Slower | Z | MA | VHL | $R_{79}$ | Far | HL | Slower | L | MB | HR |
| $R_{24}$ | Near | SR | Slower | L | HA | VHL | $R_{80}$ | Far | HL | Equal | Z | VHB | SL |
| $R_{25}$ | Near | SR | Equal | R | Z | Z | $R_{81}$ | Far | HL | Equal | L | VHB | HR |
| $R_{26}$ | Near | SR | Equal | Z | SB | SL | $R_{82}$ | Far | HL | Quicker | Z | VHB | VHR |
| $R_{27}$ | Near | SR | Equal | L | SB | VHL | $R_{83}$ | Far | HL | Quicker | L | VHB | VHR |
| $R_{28}$ | Near | SR | Quicker | Z | HB | SL | $R_{84}$ | Far | SL | Slower | R | VHA | VHR |
| $R_{29}$ | Near | SR | Quicker | L | HB | | $R_{85}$ | Far | SL | Slower | Z | VHA | VHR |
| $R_{30}$ | Near | HR | Slower | R | SA | HR | $R_{86}$ | Far | SL | Slower | L | VHA | VHR |
| $R_{31}$ | Near | HR | Slower | Z | HA | SL | $R_{87}$ | Far | SL | Equal | R | MA | HR |
| $R_{32}$ | Near | HR | Slower | L | HA | VHL | $R_{88}$ | Far | SL | Equal | Z | MA | VHR |
| $R_{33}$ | Near | HR | Equal | Z | Z | SL | $R_{89}$ | Far | SL | Quicker | R | SB | VHR |
| $R_{34}$ | Near | HR | Quicker | R | HB | MR | $R_{90}$ | Far | SL | Quicker | Z | SB | VHR |
| $R_{35}$ | Near | HR | Quicker | Z | VHB | SL | $R_{91}$ | Far | SL | Quicker | L | SB | VHR |
| $R_{36}$ | Medium | HL | Slower | R | SB | | $R_{92}$ | Far | Z | Slower | Z | VHA | SL |
| $R_{37}$ | Medium | HL | Slower | Z | MA | VHR | $R_{93}$ | Far | Z | Slower | L | VHA | HL |
| $R_{38}$ | Medium | HL | Slower | L | SA | | $R_{94}$ | Far | Z | Equal | R | VHA | MR |
| $R_{39}$ | Medium | HL | Equal | R | HB | VHR | $R_{95}$ | Far | Z | Equal | Z | VHA | MR |
| $R_{40}$ | Medium | HL | Equal | Z | MB | HR | $R_{96}$ | Far | Z | Quicker | R | Z | HL |
| $R_{41}$ | Medium | HL | Equal | L | HB | VHR | $R_{97}$ | Far | Z | Quicker | Z | Z | SL |
| $R_{42}$ | Medium | HL | Quicker | R | VHB | VHR | $R_{98}$ | Far | Z | Quicker | L | Z | HL |
| $R_{43}$ | Medium | HL | Quicker | Z | VHB | SL | $R_{99}$ | Far | SR | Slower | R | HA | |
| $R_{44}$ | Medium | HL | Quicker | L | VHB | HL | $R_{100}$ | Far | SR | Slower | Z | VHA | |
| $R_{45}$ | Medium | SL | Slower | R | VHA | HR | $R_{101}$ | Far | SR | Slower | L | VHA | |
| $R_{46}$ | Medium | SL | Slower | Z | VHA | VHR | $R_{102}$ | Far | SR | Equal | R | Z | VHL |
| $R_{47}$ | Medium | SL | Slower | L | HA | HR | $R_{103}$ | Far | SR | Equal | Z | MA | |
| $R_{48}$ | Medium | SL | Equal | R | MA | VHR | $R_{104}$ | Far | SR | Quicker | Z | SB | |
| $R_{49}$ | Medium | SL | Equal | Z | Z | VHR | $R_{105}$ | Far | SR | Quicker | L | SB | |
| $R_{50}$ | Medium | SL | Equal | L | Z | MR | $R_{106}$ | Far | HR | Slower | R | SB | VHR |
| $R_{51}$ | Medium | SL | Quicker | R | MB | HR | $R_{107}$ | Far | HR | Slower | L | HB | HR |
| $R_{52}$ | Medium | SL | Quicker | Z | MB | VHR | $R_{108}$ | Far | HR | Equal | R | VHB | |
| $R_{53}$ | Medium | SL | Quicker | L | HB | HR | $R_{109}$ | Far | HR | Equal | Z | VHB | SL |
| $R_{54}$ | Medium | Z | Slower | R | VHA | MR | $R_{110}$ | Far | HR | Equal | L | VHB | MR |
| $R_{55}$ | Medium | Z | Slower | Z | VHA | Z | $R_{111}$ | Far | HR | Quicker | R | VHB | VHR |
| $R_{56}$ | Medium | Z | Slower | L | VHA | HL | $R_{112}$ | Far | HR | Quicker | Z | VHB | VHR |

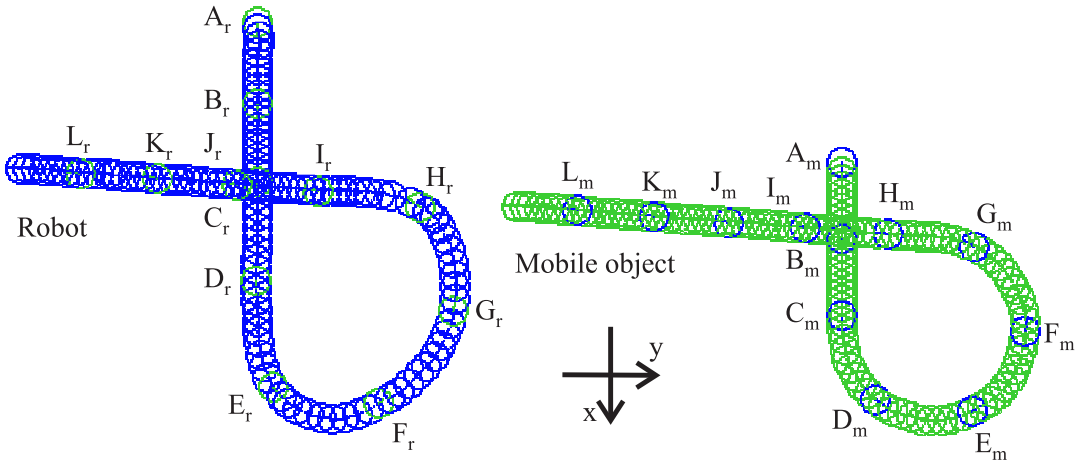
Fig. 6. Knowledge base generated by the COR-based algorithm




(a)




(b)




(c)

Fig. 7. Trajectories of the robot following different mobile objects

hand, the interpretability of the obtained rules is good, as all the linguistic labels have the same shape (triangular), are uniformly distributed along the universes of discourse, and have the same meaning for all the rules.

## V. Conclusions and future work

A methodology for the design of behaviors in mobile robotics has been presented. The main characteristics of the approach are: first, the designer only needs to define the universe of discourse, number of labels and precision of each variable, together with the scoring function ($SF$). In second place, learning is done using a set of training examples that have been automatically generated covering the whole universe of discourse of each one of the variables. This makes the learnt behavior very general, so the robot will be capable to face any situation. In third place, the learning process is very fast. Finally, the obtained knowledge base has a high interpretability, which makes easy to detect possible errors during the design or the learning process.

This methodology has been applied to the design of the "follow a mobile object" behavior. The controller with 112 linguistic rules has been tested with three complex types of trajectories for the mobile object showing good results in the average values of some parameters that reflect the quality of the behavior. In a near future, a system for the detection of moving objects will be implemented to provide the controller with the information of the object.

## Acknowledgment

This work was supported in part by the Spanish Ministry of Science and Technology under grants no. TIC2003-09400-C04-03, TIC2002-04036-C05-01, and the DXID of the Xunta de Galicia under grant no. PGIDIT04TIC206011PR.

## References

[1] R. R. Murphy, *Introduction to AI robotics*. MIT Press, 2000.

[2] A. Saffiotti, "The uses of fuzzy logic in autonomous robot navigation," *Soft Computing*, vol. 1, no. 4, pp. 180–197, 1997.

[3] M. Mucientes, R. Iglesias, C. V. Regueiro, A. Bugarín, and S. Barro, "A fuzzy temporal rule-based velocity controller for mobile robotics," *Fuzzy Sets and Systems*, vol. 134, pp. 83–99, 2003.

[4] H. Hagras, V. Callaghan, and M. Collin, "Learning and adaptation of an intelligent mobile robot navigator operating in unstructured environment based on a novel online fuzzy-genetic system," *Fuzzy Sets and Systems*, vol. 141, pp. 107–160, 2004.

[5] K. Izumi, K. Watanabe, and S.-H. Jin, "Obstacle avoidance of mobile robot using fuzzy behavior-based control with module learning," in *Proceedings of the 1999 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 1999, pp. 454–459.

[6] D. Gu, H. Hu, J. Reynolds, and E. Tsang, "Ga-based learning in behaviour based robotics," in *Proceedings of the 2003 IEEE International Symposium on Computational Intelligence in Robotics and Automation*, Kobe (Japan), 2003, pp. 1521–1526.

[7] D. Katagami and S. Yamada, "Interactive classifier system for real robot learning," in *IEEE International Workshop on Robot-Human Interaction (ROMAN-2000)*, Osaka (Japan), 2000, pp. 258–263.

[8] D. Floreano and F. Mondada, "Evolutionary neurocontrollers for autonomous mobile robots," *Neural Networks*, vol. 11, pp. 1461–1478, 1998.

[9] A. Berlanga, A. Sanchis, P. Isasi, and J. M. Molina, "A general learning co-evolution method to generalize autonomous robot navigation behavior," in *Proceedings of the 2000 Congress on Evolutionary Computation*, La Jolla, CA (USA), 2000, pp. 769–776.

[10] C. K. Lin, "A reinforcement learning adaptative fuzzy controller for robots," *Fuzzy sets and systems*, vol. 137, pp. 339–352, 2003.

[11] C. Zhou, "Robot learning with GA-based fuzzy reinforcement learning agents," *Information Sciences*, vol. 145, pp. 45–68, 2002.

[12] D. Gu, H. Hu, and L. Spacek, "Learning fuzzy logic controller for reactive robot behaviours," in *Proceedings of the 2003 IEEE/ASME International Conference on Advanced Intelligent Mechatronics (AIM 2003)*, 2003, pp. 46–51.

[13] H. R. Beom and H. S. Cho, "A sensor-based navigation for a mobile robot using fuzzy logic and reinforcement learning," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 25, no. 3, pp. 464–477, 1995.

[14] Z. Kalmár, C. Szepesvári, and A. Lörincz, "Module-based reinforcement learning: Experiments with a real robot," *Machine Learning*, vol. 31, pp. 55–85, 1998.

[15] S. Thongchai, "Behavior-based learning fuzzy rules for mobile robots," in *Proceedings of the American Control Conference*, Anchorage, AK (USA), 2002, pp. 995–1000.

[16] J. Casillas, O. Cordón, and F. Herrera, "COR: A methodology to improve ad hoc data-driven linguistic rule learning methods by inducing cooperation among rules," *IEEE Transactions on Systems, Man, and Cybernetics—Part B: Cybernetics*, vol. 32, no. 4, pp. 526–537, 2002.

[17] ——, "COR methodology: a simple way to obtain linguistic fuzzy models with good interpretability and accuracy," in *Accuracy improvements in linguistic fuzzy modeling*, J. Casillas, O. Cordón, F. Herrera, and L. Magdalena, Eds. Heidelberg (Germany): Springer, 2003.

[18] M. Mucientes and J. Casillas, "Obtaining a fuzzy controller with high interpretability in mobile robots navigation," in *Proceedings of the 13th IEEE International Conference on Fuzzy Systems*, Budapest, Hungary, 2004, pp. 1637–1642.

[19] L.-X. Wang and J. Mendel, "Generating fuzzy rules by learning from examples," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 22, no. 6, pp. 1414–1427, 1992.

[20] M. Dorigo, V. Maniezzo, and A. Colorni, "The ant system: optimization by a colony of cooperating agents," *IEEE Transactions on Systems, Man, and Cybernetics—Part B: Cybernetics*, vol. 26, no. 1, pp. 29–41, 1996.

[21] J. Casillas, O. Cordón, I. F. de Viana, and F. Herrera, "Learning cooperative linguistic fuzzy rules using the best-worst ant system algorithm," *International Journal of Intelligent Systems*, vol. 20, pp. 433–452, 2005.

[22] O. Cordón, F. Herrera, I. F. de Viana, and L. Moreno, "A new ACO model integrating evolutionary computation concepts: the best-worst ant system," in *Proceedings of the 2nd International Workshop on Ant Algorithms*, Brussels, Belgium, 2000, pp. 22–29.