Analysis of Sensitivity and Robustness to a Fuzzy System developed to guide a Simulated Robot through a Virtual World with Obstacles

Patrick B. Moratori
Instituto de Matemática / NCE
Universidade Federal do Rio de Janeiro
Post Office Box 68530, Rio de Janeiro
Brasil
moratori@posgrad.nce.ufrj.br

Márcia V. Pedro
Instituto de Matemática / NCE
Universidade Federal do Rio de Janeiro
Post Office Box 68530, Rio de Janeiro
Brasil
marciavp@posgrad.nce.ufrj.br

Emilia B. Ferreira
Instituto de Matemática / NCE
Universidade Federal do Rio de Janeiro
Post Office Box 68530, Rio de Janeiro
Brasil
emiliabf@posgrad.nce.ufrj.br

Adriano J. O. Cruz
Instituto de Matemática / NCE
Universidade Federal do Rio de Janeiro
Post Office Box 68530, Rio de Janeiro
Brasil
adriano@nce.ufrj.br

***Abstract* – This work presents a fuzzy system designed to guide a simple simulated robot through a virtual world populated with random,ly placed obstacles. The main goals were to investigate the performance of a fuzzy controller applied to the problem of moving an object in a complex environment and to study the sensitivity and robustness of this controller. The sensors used by the robot were very simple, so its knowledge of its environment was very limited. The system was quickly engineered and proved to be very successful reaching the goal in all designed tests. The tests also proved that the controller is very robust. The simplicity, easiness of design and robustness of the controller, suggests that for many control tasks, it is possible to create a fuzzy system that can perform very well under real conditions.**

I. INTRODUCTION

In this work we present a fuzzy system designed to guide a simple simulated robot through a virtual world populated with randomly placed obstacles. The main goals were to investigate the performance of the fuzzy controller applied to the problem of moving an object in a complex environment and to study the sensitivity and robustness of this controller. The sensors used by the robot were very simple, so its knowledge of its environment was very limited.

II. MODEL DESCRIPTION

Figure 1 shows the simulated robot and the virtual world. The fuzzy controller should move the robot from the left sidewall to right sidewall avoiding the fixed obstacles randomly distributed and the other sidewalls. The radius of the robot is equal to 6 units and the radius of all obstacles is equal to 3 units. The virtual world corresponded to the plane [0x200], [0x100]. The goal was to make the robot arrive at the right sidewall defined at the position $x_f = 200$ and at any coordinate y. The robot moved forward by some fixed distance at every stage of the simulation, however it did not have the capacity of rotating over its own axis.

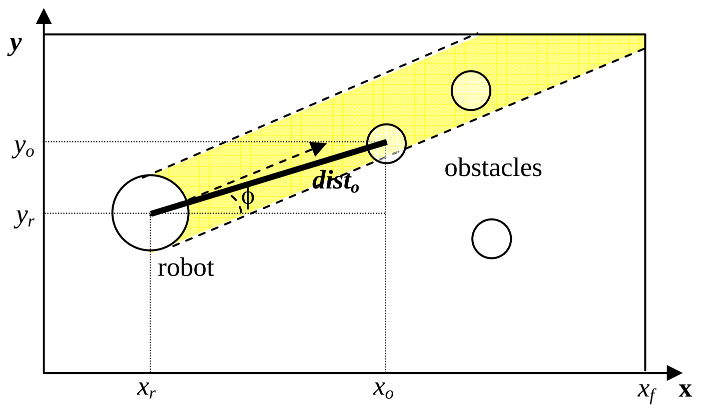

Fig. 1. Diagram of simulated robot and the virtual world

The three state variables x_r, y_r and ϕ exactly determine the robot position. The robot has sensors that detect and measure the distances to obstacles lying straight ahead in its path ($dist_o$). Figure 1 shows how the sensors are used to detect collision and measure distances. If more than one obstacle is detected only the closest one is considered. At every stage the fuzzy controller should produce the angle θ that moves the robot to the wall at the right hand side and at the same time establishes a course that avoids the obstacles and the upper and lower sidewalls.

We used simple kinematic equations similar to the ones described by Kosko and Kong [1] to the problem of backing up a truck, as in (1). The equations that describe the movement of the robot at every stage of the simulation are:

$$\begin{aligned} \phi' &= \phi + \theta \\ x'_r &= x_r + \Delta\cos(\phi') \qquad (1) \\ y'_r &= y_r + \Delta\sin(\phi') \end{aligned}$$

Δ denotes the fixed driving distance of the robot for all forward movements.

The controller's input variables are the robot angle ϕ, the y-position of the robot (y_r) and the distance to the closest obstacle $dist_o$. The output variable is the steering-angle signal θ. The variable ranges, or their universe of discourse, are:

$$\begin{gathered} 0 \le dist_o, x_r \le 200 \\ 0 \le y_r \le 100 \\ -180^o \le \phi \le +180^o \\ -30^o \le \theta \le +30^o \end{gathered} \qquad (2)$$

Positives values of ϕ and θ represent counter clockwise rotations of the robot and steering wheel respectively. Negative values represent clockwise rotations, as in (2). Therefore usual geometric signal conventions were used.

The universe of discourse of the input and output variables are divided into fuzzy sets that represent the semantic of linguistic terms, the same terms an expert would use to guide the robot through the virtual world. Fuzzy sets represent the semantic of their labels using functions that assign a real number between 0 and 1 to every element *x* in the universe of discourse *X*. The number $\mu_A(x)$ represents the degree to which the object *x* belongs to the fuzzy set A. Designers of fuzzy systems depending on their preference and experience have used many different fuzzy membership functions. In practice, triangular and trapezoidal functions are the most used because they simplify the computation and give very good results.

Figure 2 shows the membership-function graphs of the fuzzy sets defined to be used in the controller. The fuzzy sets ZE, LCW and LCCW and the fuzzy sets RLR, ZE and RLL are narrower than the other ones. These narrow fuzzy sets permit fine control near the horizontal direction, avoiding large changes when the robot is heading in the right direction. We used larger fuzzy sets at the extremes in order to obtain rough control when the robot is heading towards directions that are far from the goal. The fuzzy sets, VN, N and MD that indicate distances of obstacles closer to the robot are represented by narrower sets in order to help the controller make quick turns.

The controller, in normal conditions, operates based mainly on information from the distance to the nearest obstacle or wall (*dist$_o$*) and the angle to the horizontal (ϕ). So, this part of the rule base is composed of 35 rules that allow the robot to turn to the right as well as to the left. Table 1 shows these rules.

Table 1. Rule base matrix for the robot controller

ϕ / **dist$_o$**	**VCCW**	**CCW**	**LCCW**	**ZE**	**LCW**	**CW**	**VCW**
VN	RVR	RLL	RLL	RL	RVL	RVL	RVL
N	RVR	RLL	RLL	RL	RVL	RVL	RVL
MD	RVR	RR	RLR	RLL	RLL	RL	RVL
F	RVR	RR	RLR	RLR	RLL	RL	RVL
VF	RVR	RR	RLR	RLR	RLL	RL	RVL

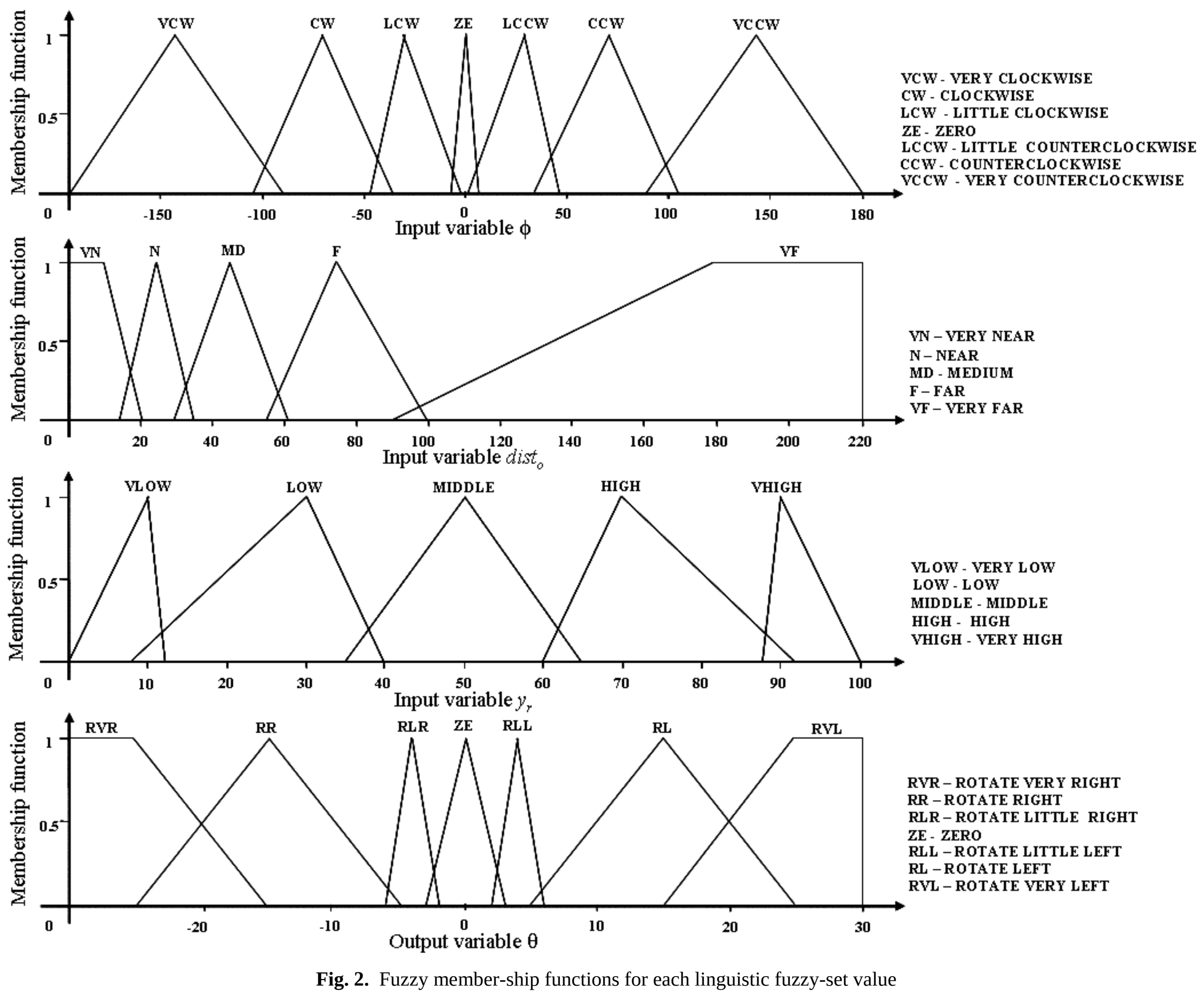

Fig. 2. Fuzzy member-ship functions for each linguistic fuzzy-set value

There are some particular situations, when the robot is close to the upper or lower walls that the rules shown above do not guide it correctly. In order to solve this problem we have two rules that take into account the *y*-position of the robot. These rules use only two fuzzy sets from the y_r variable and are independent of the other two input variables, as in (3).

$$\begin{array}{c}\text{if } (y_r \text{ is VLOW) then } (\theta \text{ is RVL}) \\ \text{if } (y_r \text{ is VHIGH) then } (\theta \text{ is RVR})\end{array} \qquad (3)$$

III. TESTING SENSITIVITY

First the original controller was tested in different situations in order to evaluate its ability to guide the robot through the virtual world. In these experiments, the number of obstacles was varied between 1 and 5, randomly distributed over the virtual world. For each of these arrangements we tested different starting positions (10 positions) and starting angles (5 angles) evenly distributed over the universe of discourse of each variable. Note that the robot always started at the left sidewall. The controller was able to guide the robot correctly to its goal in all of these randomly generated situations.

In order to test the sensitivity of the controller to the removal of rules from the rule base we chose from the previous experiment a distribution of the obstacles that guarantee that the robot will always have to avoid at least one obstacle independent of its initial position. The coordinates of the five obstacles are shown in the Table 2.

Initially we defined 40 initial positions, combining initial values of ϕ (-90º, -45º, 0º, 45º, 90º) and y_r (15, 25, 35, … , 85). Each one of these 40 initial positions, that we called a sample, was processed first using the complete controller. At each step of an experiment one rule was removed, according to predefined criterion, and all 40 initial configurations were reprocessed. The reprocessing ends when all the rules are removed.

The figure 3 shows an example of a complete trajectory of the robot starting at $y_{r=}$ 7 and ϕ = -30º.

Table 2. Coordinates of the five obstacles.

X_o	Y_o
59.1	48.5
78.8	19.4
118.2	77.6
137.9	29.1
157.6	58.2

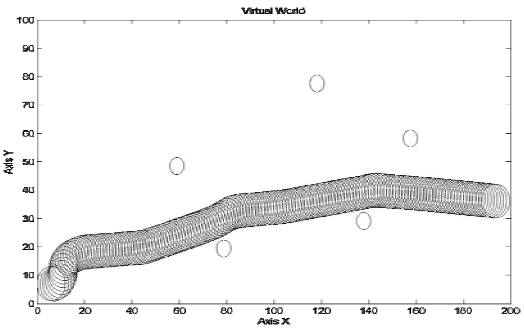

Fig. 3. Example of a robot trajectory

We defined a total of 14 different ways of removal of rules. Of these ways, ten used random removal and the other four used the following order of removal: (1) sequential removal using column-wise order; (2) reverse sequential column-wise, meaning that the rules from the last column will be removed first; (3) central rules first (shaded cells in the Table 1) and then peripheral rules; (4) peripheral rules first then central rules. Combining all these different possibilities we had a total of 20720 experiments.

IV. SIMULATIONS AND RESULTS

We analysed the sensibility of the controller in two ways. First we used the travelling error, which was defined as the Euclidean distance from the actual final position x_r to the desired final position x_f. Note that the robot stops when it crashes into a wall or an obstacle and that it is considered a fail.

The average travelling error of all experiments is shown in Figure 4 (a). Additionally is shown the standard deviation (b), in order to estimate the dispersion of the average.

There were some particular situations where the robot was not able to achieve its final goal from any of the initial positions; however the error average was small when compared to other samples. In order to analyse these situations we measured the quantity of samples that actually achieved the desired goal. Remember that the goal was to achieve the left wall independent of its trajectory or time spent. Figure 4 (c) shows the quantity of samples that achieved the left wall as the rules were removed.

Figure 4 (c) shows that 79.64 % of all samples were able to achieve the goal even after 10 rules are removed. After 20 rules were removed, approximately 34.46 % of the samples still worked correctly.

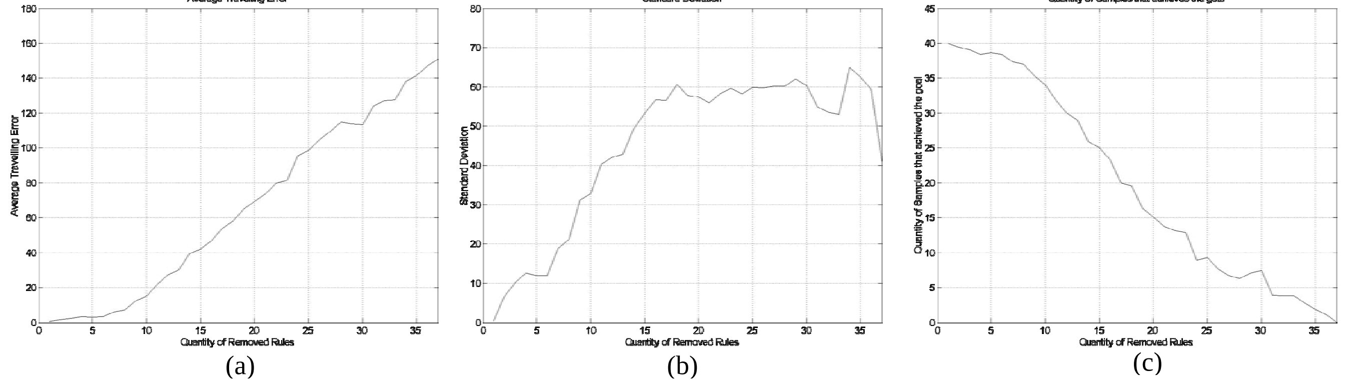

Fig. 4. Average travelling error (a), standard deviation (b) and quantity of samples that achieved the goal (c).

Figure 5 shows the quantity of samples that achieved the goal when the reverse sequential column-wise order of removal was used. There is a sharp decrease of the quantity of samples that achieve the goal after the removal of 21 rules. This behaviour did not repeated itself when the same rule was removed in other experiments and this same pattern happened for other rules. This behaviour suggests that the importance of a rule is determined by the order and the quantity of the rules already removed. It is interesting to note in the Figure 5 that after the next rule was removed (22 rules) the number of positive experiments increased again. Therefore, some rules have an inhibitory (or positive) effect on some behaviour and after its removal the robot was free (impeded) to make a particular set of decisions.

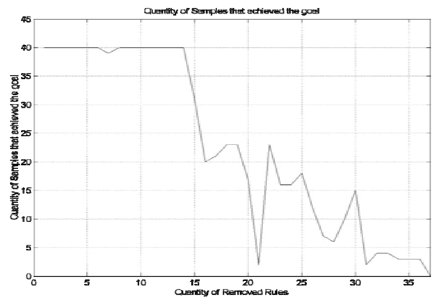

Fig. 5. Quantity of samples that achieved the goal according to a defined criterion of removal of rules

Another interesting analysis refers to the order of removing two important groups of rules: peripheral and central rules. Figure 6 (a) shows the quantity of samples that achieved the goal when peripheral rules are removed first. Figure 6 (b) shows the results when the central rules are removed first.

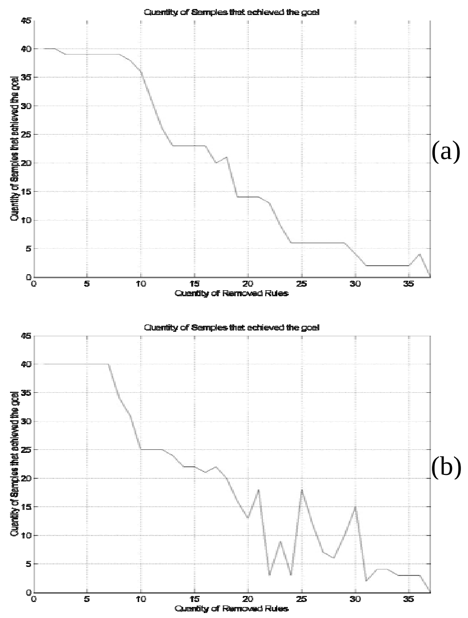

Fig. 6. Quantity of samples that achieved the goal according to a defined criteria of removal of rules

From the figure it is possible to see that both strategies present approximately the same quantity of hits, therefore it is possible to conclude that the central rules and the peripheral ones have the same importance to the system.

V. CONCLUSIONS

In this work we tested the sensitivity of a fuzzy controller used to guide a robot through a virtual world. The controller was very simple and needed very few sensors to acquire all the necessary information. The system was quickly engineered and proved to be very successful reaching the defined goal in all designed tests. The controller still performed very well even after 27% of the rules were removed, considering that in approximately 80% of all tests it still reached the goal. The tests indicate that peripheral and central rules have the same importance since the order in that they were removed had little influence in the sensitivity of the system. This simplicity, easiness of design and robustness, suggests that for many control tasks, it is possible to create a fuzzy system that can perform very well under real conditions.

VI. ACKNOWLEDGMENT

The authors gratefully acknowledge the contributions of Instituto de Matemática / Núcleo de Computação Eletrônica – Universidade Federal do Rio de Janeiro for the accomplishment of this research providing scholarships for a member of the group.

VII. REFERENCES

[1] B. Kosko and S. Kong, “Comparison of Fuzzy and Neural Truck Backer-Upper Control Systems”, in “Neural Networks and Fuzzy Systems: A Dynamical Systems Approach to Machine Intelligence” B. Kosko, Prentice Hall, Englewod Cliffs, NJ: 1992, p.339.

[2] B. Kosko, “Fuzzy Engineering”, Prentice Hall, Upper Saddle River, NJ: 1997.

[3] D. Dubois and H. Prade, “Fuzzy Sets and Systems: Theory and Applications”, Academic Press, Orlando, FL: 1980.

[4] D. Nguyen and B. Widrow, “The Truck Backer-Upper: An Example of Self-Learning in Neural Networks”, Proceedings of International Joint Conference on Neural Networks (IJCNN-89), vol. II, June 1989, pp. 357-363.

[5] E. Mamdani, “Application of Fuzzy Logic to Approximate Reasoning Using Linguistic Synthesis”, IEEE Transations on Computers, vol. C-26, nº. 12, Dec. 1997, pp. 1182-1191.

[6] J. R. Jang, C. T. Sun, E. Mizutani, “Neuro-Fuzzy and Soft Computing: A Computational Approach to Learning and Machine Intelligence”, Prentice Hall, Englewod Cliffs, NJ: 1996.

[7] J. Yen, R. Langari, “Fuzzy Logic: Intelligence, Control and Information”, Prentice Hall, Englewod Cliffs, NJ: 1999.