Computational Intelligence and Decision Making

Trends and Applications

Springer

Chapter 29
Development of an AGV Controlled by Fuzzy Logic

Ramiro S. Barbosa, Manuel F. Silva, and Dário J. Osório

Abstract This paper presents the development of a behavior-based AGV using fuzzy logic. A robot platform and a fuzzy logic controller (FLC) are developed for the embodiment of different behaviors. Experimental results are given to assess the performance of the AGV and to validate the proposed design schemes for its construction and control.

29.1 Introduction

Fuzzy logic has emerged in the 1960s, more precisely in 1965, when the first article was published by Zadeh. However, only in the following decade were developed the first applications in the field of automatic control by Mamdani. The control by fuzzy logic allows a different approach to the problem, in which intuitive knowledge on how to best control the process should be acquired, and this information will be part of the FLC. FLCs have been successfully applied in the control of many physical systems, particularly those with uncertainty, unmodelled, disturbed and/or with nonlinear dynamics [1, 2]. Nowadays, one of the main applications of FLC is in the area of autonomous robotics, where several works on behavior-based fuzzy control have been developed [3–5].

Bearing these ideas in mind, this paper presents the development of a mobile robot, with an open and distributed architecture, controlled by fuzzy logic, and capable for the embodiment of different behaviors. The rest of the paper is organized as follows. Section 29.2 addresses the design of the AGV. Section 29.3

R.S. Barbosa (✉) • M.F. Silva • D.J. Osório
GECAD – Knowledge Engineering and Decision Support Research Center,
Institute of Engineering – Polytechnic of Porto (ISEP/IPP), Porto, Portugal
e-mail: rsb@isep.ipp.pt; mss@isep.ipp.pt; darioosorio@msn.com

A. Madureira et al., *Computational Intelligence and Decision Making: Trends and Applications*, Intelligent Systems, Control and Automation: Science and Engineering 61,
DOI 10.1007/978-94-007-4722-7_29,

is concerned with the design of the fuzzy controller and the implementation of the several behaviors. Section 29.4 presents experimental results showing the effectiveness of the proposed design schemes for the construction and control of the robot. Finally, Sect. 29.5 draws the main conclusions.

29.2 Design of the AGV

29.2.1 AGV Structure

The robot is built adopting a modular structure (Fig. 29.1, left), being constituted by a rigid PVC base structure, two DC motors with encoder in the front, four ultrasonic sensors (also called sonars) and a free-wheel in the rear, to support part of the robot weight. The motors can develop a maximum torque of 6.12 kg/cm and a maximum velocity of 66 rpm. The sonars are of the type SRF05, with range from 3 cm to 4 m. Three microcontrollers of the PIC 18F family are used, namely the 18F4585, with a CAN interface for the communication between the several modules of the system. Among other characteristics, the 18F4585 possesses timers and A/D converters, can

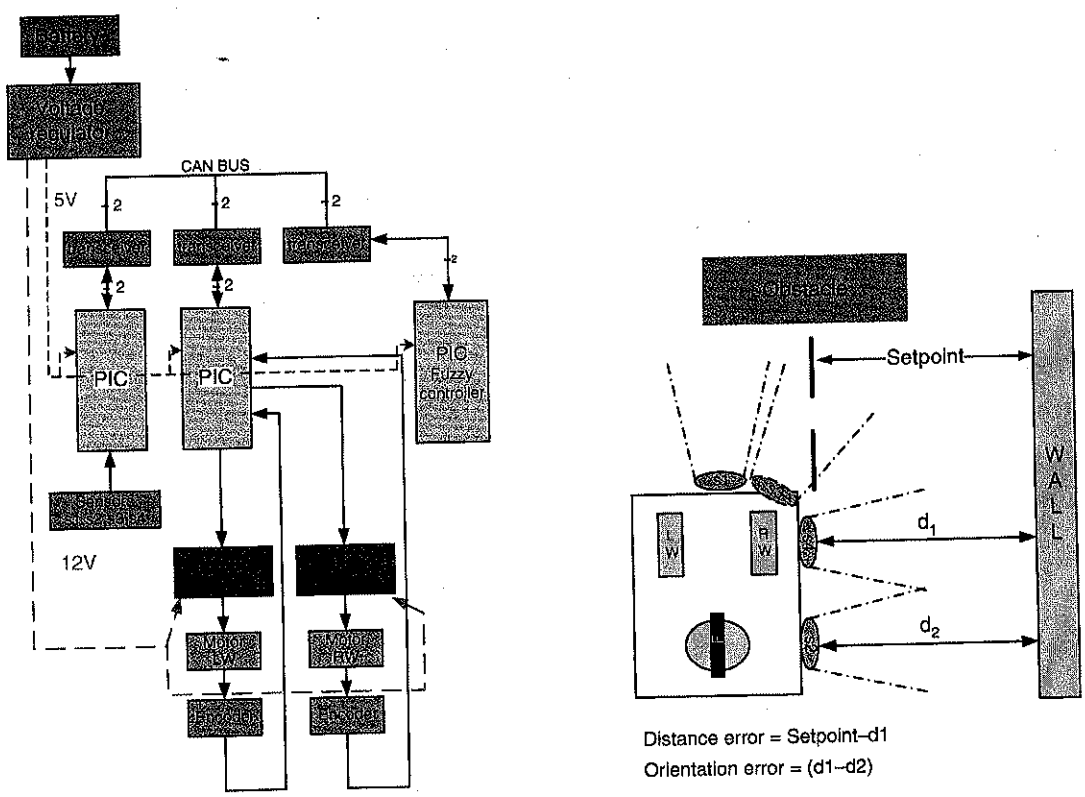

Fig. 29.1 AGV architecture (*left figure*) and schematic representation of the AGV physical structure (*right figure*)

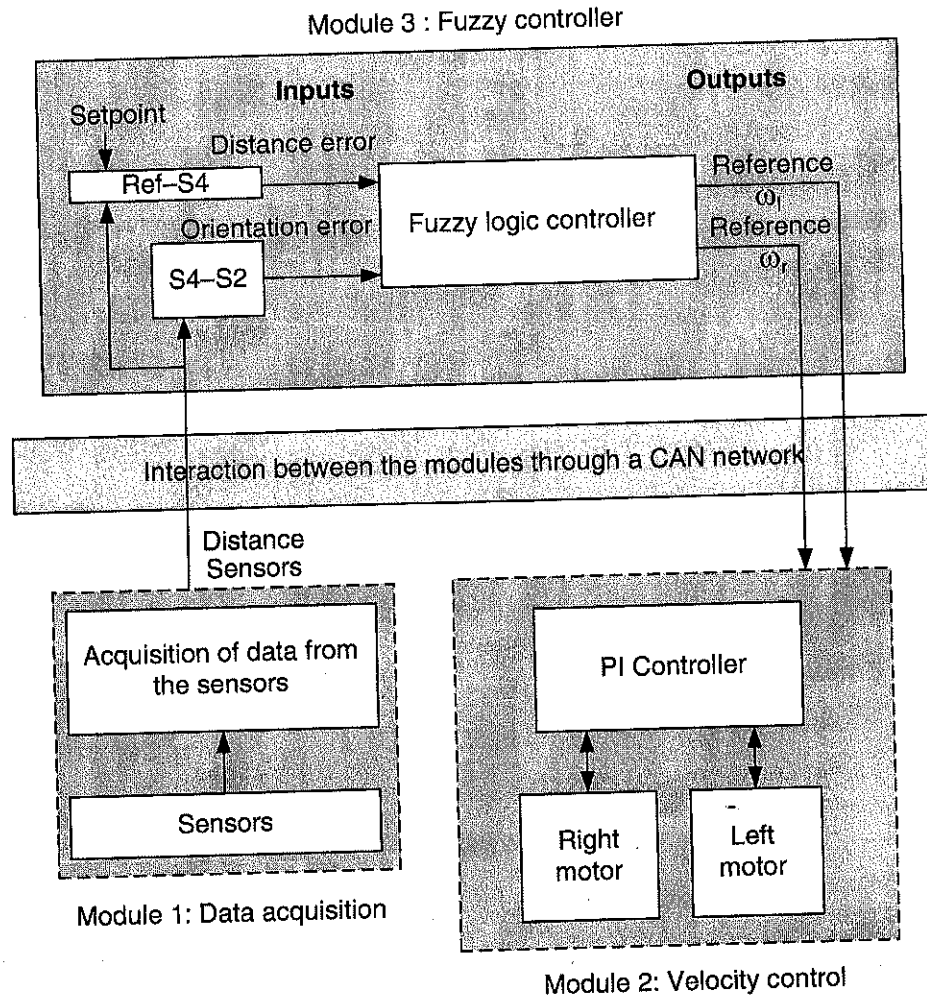

Fig. 29.2 Features of the modules of the AGV and interaction between them

operate at frequencies up to 40 Mhz and presents 48 kbytes of program memory, 3,328 bytes of data memory and 1,024 bytes of EEPROM.

The schematic representation of the AGV physical structure is presented in Fig. 29.1 (right). The two right side sonars (S2 and S4) are used to determine the orientation and the distance of the robot from the wall. This configuration of the sonars is used for the implementation of the right wall-following behaviour. The front sonars (S1 and S3) allow the detection of obstacles and the implementation of the emergency behaviour.

Figure 29.2 illustrates a block diagram showing the features implemented in the three modules that constitute the AGV and the interaction between them. Module 1 performs the data acquisition from the sonar sensors. Module 2 is responsible for the velocity control of the left wheel (LW) and right wheel (RW) differential traction allowing orienting the robot in space. Finally, Module 3 implements the FLC. Each of these modules has an interface for communication according to the CAN protocol. The use of the CAN bus in this work aimed not only to distribute the several tasks across the multiple nodes, but was also chosen due to the flexibility that this network offers in the incorporation of future modules into the system.

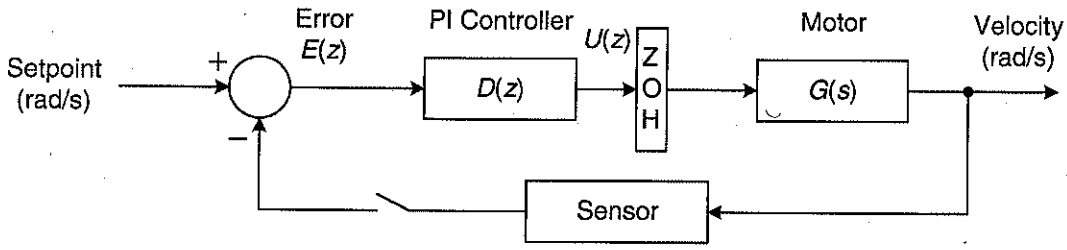

Fig. 29.3 Block diagram of the system with the PI controller

29.2.2 Velocity Control

Figure 29.3 illustrates the block diagram of the digital control system with the PI controller used for the velocity control of both motors (RW and LW) of the AGV.

The transfer function between the output velocity $\omega(s)$ and the applied voltage $U(s)$ of the set motor plus actuator is given by the first-order model [6]:

$$G(s) = \frac{\omega(s)}{U(s)} = \frac{6.72}{0.0454s + 1} \tag{29.1}$$

The digital PI controller was designed through the discrete root-locus [7], yielding:

$$D(z) = \frac{U(z)}{E(z)} = K_c \frac{z - \alpha}{z - 1} \tag{29.2}$$

with $K_c = 0.0472$, $\alpha = 0.2869$, and a sampling period of $T = 0.01$ s.

29.2.3 AGV Kinematics

Figure 29.4 shows a scheme of the kinematic model of the robot, where $v_r(t)$ is the linear velocity of the right wheel, $v_l(t)$ is the linear velocity of the left wheel, $\omega_{system}(t)$ and $V_{system}(t)$ are respectively the angular and linear velocities of the robot, θ is the orientation angle of the system, and L is the distance between the two wheels.

The linear velocity for each motor is calculated using the relation $v = \omega r$ where r is the radius of the wheel. The linear and angular velocities of the robot system, V_{system} and ω_{system}, respectively, can be obtained through the equations:

$$V_{system} = \frac{v_r + v_l}{2}, \quad \omega_{system} = \frac{v_r - v_l}{L} \tag{29.3}$$

where r and L are 3.5 and 21.7 cm, respectively.

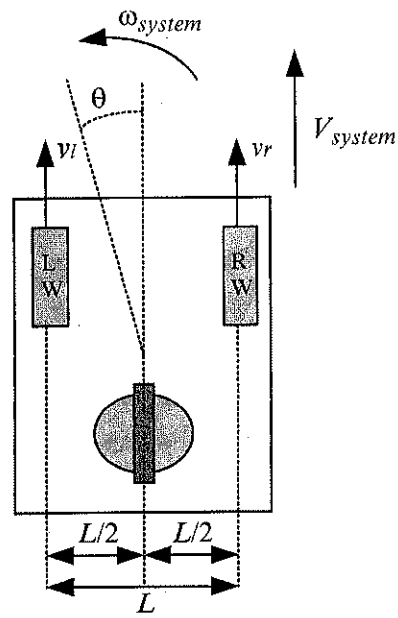

Fig. 29.4 AGV kinematics

29.3 Design of the Fuzzy Controller

29.3.1 Behaviors

The robot platform is used to incorporate right wall-following, avoid-obstacles and emergency behaviors, as illustrated in Fig. 29.5. For the wall-following behavior, the input variables are the distance error from the wall (*ErrorDistance*) and the orientation error of the robot (*Orientation*), determined from readings of the side sonars S2 and S4 (Fig. 29.1, right). For the avoid-obstacles and emergency behaviors are used the readings of the robot front sonars (S1 and S3) (Fig. 29.1, right). On the other hand, the outputs are the linear velocity (*LinearVelocity*) and the angular velocity (*AngularVelocity*), which will determine the velocities to be applied to each motor of the AGV.

29.3.2 Wall-Following Behavior

Figure 29.6 shows the membership functions associated with the input and output variables for the wall-following behavior. For each fuzzy input five membership functions are constructed: *BigRight*, *Right*, *Zero*, *Left* and *BigLeft* (Fig. 29.6, left). The output *LinearVelocity* is characterized by four membership functions (*Zero*, *Low*, *Midle* and *High*) while the output *AngularVelocity* is defined through five membership functions: *BigRight*, *Right*, *Zero*, *Left* and *BigLeft*, as illustrated in Fig. 29.6 (right).

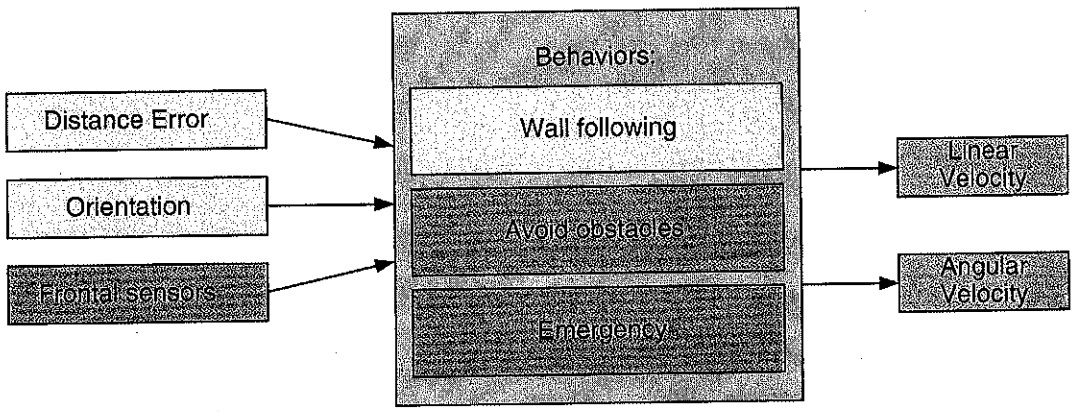

Fig. 29.5 Designed and implemented behaviors

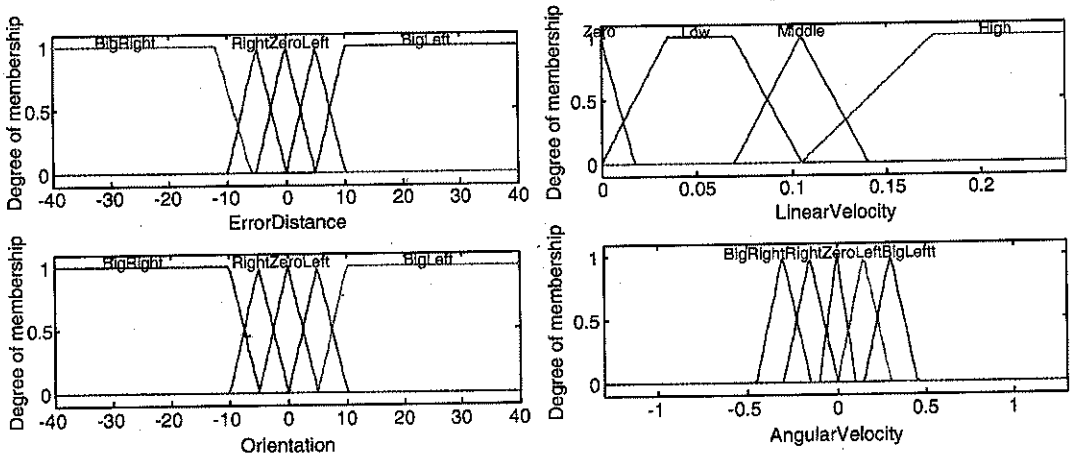

Fig. 29.6 Membership functions of input and output variables for the wall-following behavior

The rule bases containing the control fuzzy rules for the proposed FLC are listed in Tables 29.1 and 29.2, for the outputs *LinearVelocity* and *AngularVelocity*, respectively.

29.3.3 Implementation

The fuzzy implication and the aggregation of the rules were implemented by using the *min* and *max* operators, respectively. The centroid method was used for the defuzzification of the output fuzzy sets. Since, ω_{system} is calculated as in (29.3), the difference of velocities between the motors is $\Delta v = \omega_{system} L$. Since the desired linear velocity is already known, the velocities to be applied in each motor are given by:

$$v_r = V_{system} - \frac{\Delta v}{2}, \quad v_l = V_{system} + \frac{\Delta v}{2} \tag{29.4}$$

Table 29.1 Rule base of output variable *LinearVelocity* for the wall-following behavior

Linear velocity		ErrorDistance				
		Big Left	Left	Zero	Right	Big Right
Orientation	Big Right	Middle	Middle	Middle	Middle	Middle
	Right	Middle	Middle	Middle	Middle	Middle
	Zero	Middle	Middle	Middle	Middle	Middle
	Left	Middle	Middle	Middle	Middle	Middle
	Big Left	Middle	Middle	Middle	Middle	Middle

Table 29.2 Rule base of output variable *AngularVelocity* for the wall-following behavior

Angular velocity		ErrorDistance				
		Big Left	Left	Zero	Right	Big Right
Orientation	Big Right	Big Left	Left	Left	Zero	Zero
	Right	Left	Left	Zero	Left	Zero
	Zero	Left	Zero	Zero	Zero	Right
	Left	Zero	Right	Zero	Right	Right
	Big Left	Zero	Zero	Right	Right	Big Right

The resultant linear velocities are divided by the radius of the corresponding wheel and their values are sent, through the CAN bus, to Module 2 (Velocity control).

29.3.4 Avoid-Obstacles Behavior

The avoid-obstacles behavior helps the robot move freely without crashing against objects. This behavior uses the readings of the two front sonars (S1 and S3, see Fig. 29.1, right). Figure 29.7 shows the membership functions associated with the input and output variables. In order to improve the obstacles avoidance capability of the AGV, two new terms were added in the definition of the output variable *AngularVelocity* (*BigBigRight* and *BigBigLeft*). The corresponding rule bases are presented in Tables 29.3 and 29.4.

29.3.5 Emergency Behavior

The emergency behavior is fundamental for the operation of the robot, since it guarantees a safe distance between the robot and the objects. This behavior results from a particular set of rules that are not considered in the avoid-obstacles behavior. Table 29.5 presents the actions that should be considered in the emergency behavior.

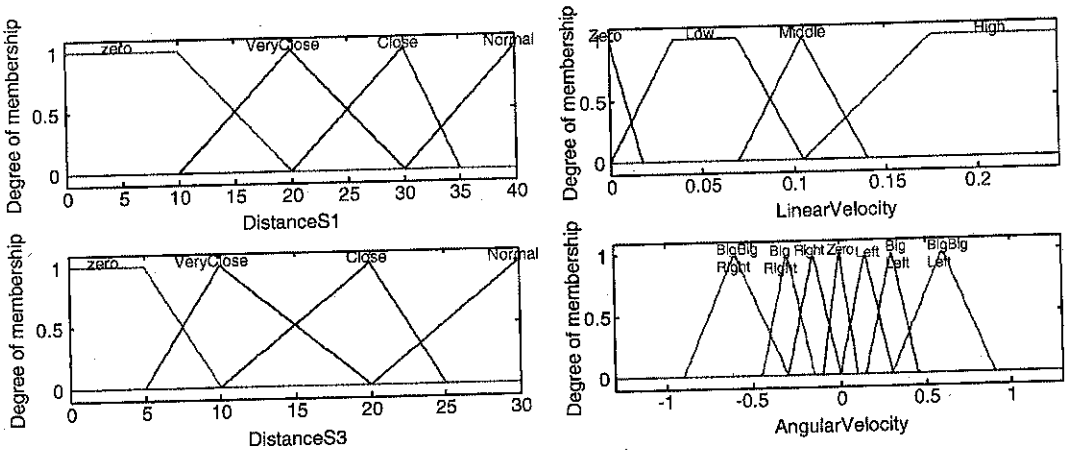

Fig. 29.7 Membership functions of input and output variables for the avoid-obstacles behavior

Table 29.3 Set of rules of output variable *LinearVelocity* for the avoid-obstacles behavior

Linearvelocity		Sensor S1 – DistanceS1			
		Zero	VeryClose	Close	Normal
Sensor S3 – DistanceS3	Zero	Emergency behavior	Emergency behavior	Emergency behavior	Emergency behavior
	VeryClose	Emergency behavior	Middle	Middle	Middle
	Close	Emergency behavior	Middle	Middle	Middle
	Normal	Emergency behavior	Middle	Middle	Wall following behavior

Table 29.4 Set of rules of output variable *AngularVelocity* for the avoid-obstacles behavior

Angular velocity		Sensor S1 – DistanceS1			
		Zero	VeryClose	Close	Normal
Sensor S3 – DistanceS3	Zero	Emergency behavior	Emergency behavior	Emergency behavior	Emergency behavior
	VeryClose	Emergency behavior	BigBigLeft	BigBigLeft	BigBigLeft
	Close	Emergency behavior	BigBigLeft	BigBigLeft	BigLeft
	Normal	Emergency behavior	BigBigLeft	BigLeft	Wall following behavior

Table 29.5 Set of rules of output variables *LinearVelocity* and *AngularVelocity* for the emergency behavior

		Sensor S1 – DistanceS1			
		Zero	VeryClose	Close	Normal
Sensor S3 – DistanceS3	Zero	Zero	Zero	Zero	Zero
	VeryClose	Zero	Avoid obstacles behavior	Avoid obstacles behavior	Avoid obstacles behavior
	Close	Zero	Avoid obstacles behavior	Avoid obstacles behavior	Avoid obstacles behavior
	Normal	Zero	Avoid obstacles behavior	Avoid obstacles behavior	Wall following behavior

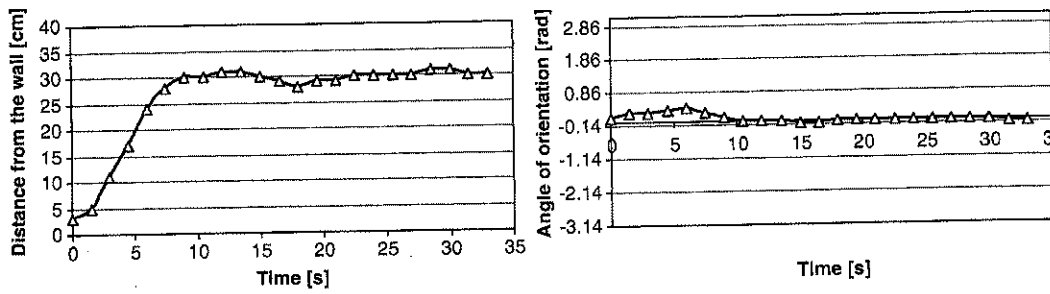

Fig. 29.8 Evolution of the distance from the wall (*left figure*) and of the angle of orientation of the robot (right figure)

Since different behaviors are implemented in the same FLC, it is necessary to define an arbitration method to coordinate the activation of the different behaviors. The arbitration was done by a set of rules that define different contexts. These context rules make the management of the active behaviors in order to guarantee that no conflicts occur in the output actions. The management of the behaviors is performed by a set of rules with the format **IF** *DistanceS1* is *Zero* **Or** *DistanceS3* is *Zero* **Then** *Emergency Behavior*.

If no conflicts exist in the output fuzzy sets, the defuzzification process can be performed in order to get the control values to the system. Note that these context rules, that make the coordination of the different behaviors, guarantee that only the rules of the active behaviors are processed, so some processing time is saved.

29.4 Experimental Results

In order to verify the performance of the robot system, several experimental tests were performed [6]. Figure 29.8 shows the evolution of the distance from the wall and the angle of orientation of the robot for a setpoint (desired distance from the wall) of 30 cm. As can be seen, the robot starts at a distance from the wall of approximately zero (near to the wall) and tends to follow the wall keeping the desired distance.

After the implementation of all behaviors in the FLC, a test was realized that allowed analyzing the behavior of the robot in different situations. Fig. 29.9 shows the environment where the robot was tested (left) and presents the experimental results obtained from the experiment (right). In the first instants the robot tries to reach the setpoint of 30 cm from the wall; then (at around 10 s) an obstacle is detected and the robot turns left, avoiding it. When the obstacle is no longer at the right side of the robot, it turns right until detecting the wall. Once the robot detects the wall, it considers that the wall is an obstacle and turns left again until staying parallel to it. This test allowed validating the system, since the robot was capable to traverse the path, in the desired way, without colliding with the obstacle.

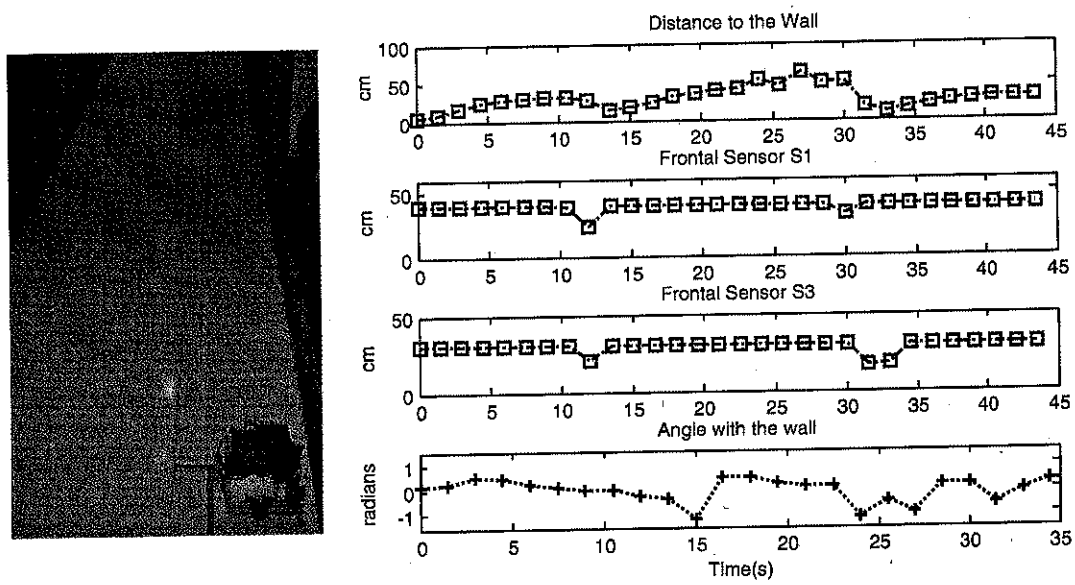

Fig. 29.9 Environmental conditions (*left figure*) and experimental results (*right figure*)

29.5 Conclusions

This paper presented the development of an AGV, with an open and distributed architecture, controlled by fuzzy logic, and capable for the embodiment of different behaviors. It was verified that the robot performs as expected, assessing the design options adopted for its construction and control. The vehicle was designed to have mainly reactive features, in order to follow walls and avoid obstacles. It was verified that the robot is able to perform these tasks with success.

References

1. Passino KM (1998) Fuzzy control. Addison-Wesley, Menlo Park
2. Reznik L (1997) Fuzzy controllers. Newnes, Oxford
3. Saffiotti A (1997) The uses of fuzzy logic in autonomous robot navigation. Soft comput 1:180–197
4. Thongchai S, Suksakulchai S, Wilkes DM, Sarkar N (2000) Sonar behaviour-based fuzzy control for a mobile robot. In: Proceedings of the IEEE international conference on systems, man, and cybernetics, Brisbane, Australia, 8–11 Oct 2000, p 16
5. Zein-Sabatto S, Sekmen A, Koseeyaporn P (2003) Fuzzy behaviors for control of mobile robots. J Syst Cybern Inform 1:68–74
6. Osório DJ, Barbosa RS, Silva MF (2011) Fuzzy control architecture for a mobile robot system. In: Proceedings of the 31st IASTED international conference on modelling, identification, and control, Innsbruck, Austria, 14–16 Feb 2011, pp 286–293
7. Franklin GF, Powell JD, Workman M (1998) Digital control of dynamic systems. Addison-Wesley, Menlo Park