Experimenting fuzzy control strategies for mobile robots on a rapid prototyping platform

F. CUPERTINO, L. DELFINE, V. GIORDANO, D. NASO, B. TURCHIANO
Dipartimento di Elettrotecnica ed Elettronica
Politecnico di Bari, Bari
ITALY

Abstract: - This paper describes a rapid prototyping platform developed to address the complex design of real-time decision and control strategies for commercially-available mobile robots. The platform consists in a microprocessor board fully controllable in Matlab/Simulink object-oriented programming environment, directly connected through a serial link and a specific communication protocol to the mobile robot. The architecture also encompasses a vision sensor that can be used as a further source of feedback for real-time control, allowing the users to develop, test and compare directly on the hardware navigation strategies of considerable complexity. To show the practical advantages of the platform, this paper illustrates the design of an effective behaviour-based fuzzy control strategy guaranteeing simultaneously obstacle-avoidance and target-reaching performance in an unknown environment.

Key-Words: - Mobile robots, fuzzy control, rapid prototyping, autonomous navigation.

1 Introduction

The major challenge of autonomous robotics is to build robust control schemes that reliably perform complex tasks in spite of environmental uncertainties and without human intervention. In the last decade, a great amount of research has been devoted to increase the autonomy of mobile robots and several advanced control algorithms have been proposed to guarantee successful navigation in real-world applications [3, 7-14]. The real performances of mobile robots are intimately linked to the (generally unknown) operating environment, so that design procedures based on the usual sequential schema including off-line modelling, simulation-based debugging and validation, final implementation on hardware and subsequent trial and error adjustment, seldom produces satisfactory results without compromising the actual developing times. Moreover, in many cases, developing an accurate model of the overall system for simulation analysis could be time consuming or even prohibitive, due to the uncertainty and complexity of robot mechanics, sensors, and environment. An experimental validation is also recommendable to include adverse observation conditions (e.g. poor lighting, noise defective hardware). For such reasons, even though part of recent literature still carries on design and validation of considerably sophisticated control algorithms for autonomous robot navigation only basing on simulation and idealized environments [9, 10, 11, 13], the role of experimental validation during the design is becoming more and more crucial for successful design. This, in turn, determines a further complication of prototypes development, especially in the case of advanced control architectures, since tangled hardware and software issues have to be addressed.

The need for design and verification methodologies simultaneously increasing the reliability of a design project and reducing the installation times is felt as a crucial problem in most research areas of engineering [1]. Sharing this motivation, this paper describes a flexible and modular platform to rapidly develop and experiment control algorithms for the autonomous navigation of mobile robots. Our platform has the following aims:

- allowing real-time control of commercially-available mobile robots, overcoming the inherent limitation of low-cost hardware (sensors, actuators), and improving the real-world performances;
- exploiting the potentialities of hi-level programming environment, such as the widespread and versatile Matlab/Simulink software package;
- developing a user-friendly interface for real-time monitoring and parameter tuning;
- guaranteeing modular hardware and software architectures for straightforward modifications, debugging, and evolution of the design schemes.

To give an idea of the practical advantages allowed by the proposed experimental platform, this paper describes the design of a fuzzy control architecture for autonomous robot navigation in an unknown, dynamic environment. The design goal is to find the proper path to reach a certain target avoiding all the obstacles found in the unknown environment. The adoption of fuzzy logic (FL) [3, 9-14] allows us to overcome the difficulties of properly modelling the unstructured and dynamically changing surrounding environment, hardly expressible through mathematical equations. However, while on the one hand FL allows us to directly program the decision algorithm using linguistic information and rules of thumbs, on the other one, the lack of universal and reliable design guidelines and the explicit

reliance on trial-and-error adjustments make the "simulate-and-then-experiment" approach laborious.

Thanks to the proposed rapid prototyping tool, we succeed to design a hierarchical fuzzy controller in which two different behaviours (avoid obstacles and reach the target) are designated to two distinct algorithms (based on the reactive paradigm [2, 6]), which are subsequently fused in a single control law by means of a third fuzzy controller guaranteeing the safety of the robot and the accomplishment of the task. Each layer of the hierarchical controller has been designed and tested separately before being used in the final architecture. The possibility to develop modular design of complex control strategies is a further advantage offered by the presented platform.

2 Overview of the proposed architecture

Our rapid prototyping platform is based on the widespread Khepera II robot [14]. Khepera II is a differential drive mobile robot developed by the EPFL (Ecole Polytechnique de Lausanne). It is an efficient tool for rapid, initial tests of innovative experiments, ranging from autonomous navigation to studies of communities of biological entities in real world environments [7, 8, 11]. A Khepera has two wheels, each driven by a dc motor through a 25:1 reduction gear. An incremental encoder on the motor axis gives 24 pulses per revolution of the motor that corresponds to 12 pulses per millimetre of path of the robot. The built-in motor controller accepts either position or speed commands. Eight infrared sensors are distributed around the robot for obstacle detection, although they generally provide noisy and scarcely reliable signals that are strongly influenced by lighting conditions and obstacle's reflective properties.
The readings of the encoders, of the proximity sensors and the assignment of the position/speed commands are externally available thanks to the serial link.
The Khepera can be wire-controlled with every strategy that can withstand the limited speed of serial communication, since the wire-link between Khepera and terminal allows the complete control of the functionalities of the robot through an RS232 serial line and protocol. Alternatively, it is possible to download the control code in robot's flash memory, even if this solution makes intensive trial-and-error tests more complicated (a new download is necessary for every modification of the control code).

Our interface allows to directly develop control programs for the Khepera using Matlab/Simulink codes. This relieves the designer from most of low-level hardware and software problems, letting him focus on the control system design using hi-level and intuitive libraries of toolboxes, thus fully overcoming the tedious transition from simulations to experiments. The core-equipment of our Matlab-to-Khepera interface is the dSpace dS1104 microprocessor board, since it possesses various interesting features suitable for our purposes. Namely, the board is fully programmable in Matlab/Simulink environment through Real-Time Workshop (RTW), allowing the real-time communication between the board and Matlab routines concurrently running on the PC thanks to the mlib/mtrace software. Furthermore, the design of user friendly control panels and of virtual instruments for on-line monitoring and on the fly parameter tuning can be intuitively realized with the software Control Desk [4]. Finally, and most importantly for our application, it is endowed with a RS232 serial port whose control can be realized directly via suitable real time interface Simulink blocks exploiting the ASCII based communication protocol provided by K-Team. Every interaction between terminal and Khepera is composed by:
- a command, beginning with one or two ASCII capital letters (representing specific commands for the Khepera) and followed, if necessary, by numerical or literal parameters separated by a comma and terminated by a carriage return (CR) or a line feed (LF), sent by the terminal to the Khepera robot;
- a response, beginning with the same one or two ASCII letters of the command but in lower-case and followed, if necessary, by numerical or literal parameters separated by a comma and terminated by a carriage return and a line feed, sent by the Khepera to the terminal.

For example, if we want to set the wheel speeds to +15 and -5 pulses/10ms respectively, the corresponding command to be translated in ASCII format is D,+15,-5CR. In order to increase the sensor equipment of the robot and improve design possibilities, a web cam with a top-view of the robot arena has been added to the system and connected to the PC with an USB interface. The overall experimental set-up is shown in fig. 1. The continuous lines represent physical connections between hardware units, whereas the dotted lines represent data exchanges between software routines in concurrent execution.

The mobile robot is connected with a serial cable to the dSpace controller board (mounted in one PCI slot of a PC), which executes in real time the Simulink program controlling the robot and managing the serial communication.

The RGB images provided by the web cam are processed through a Matlab routine based on colour detection. The extracted information are then continuously sent to the control scheme running on the dSpace board through the mlib/mtrace software. Since the vision algorithm runs in Matlab under Windows operative system, strict real time performances can't be guaranteed. Nevertheless, given the limited speed of the Khepera, this limitation is not particularly significant. In particular, setting the sampling time of the vision system T_v sufficiently higher than the average time to process a single image, we guarantee that the information sent by camera will be always available on time. For our experiments, we choose $T_v = 200$ ms whereas the sampling time of the control system runs at $T_c = 20$ ms.

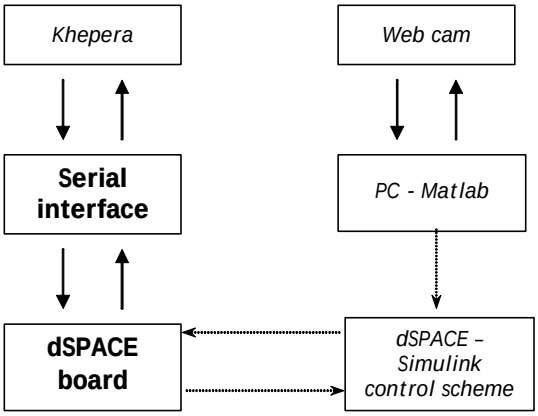


Fig. 1-Overall configuration of the proposed test bed architecture

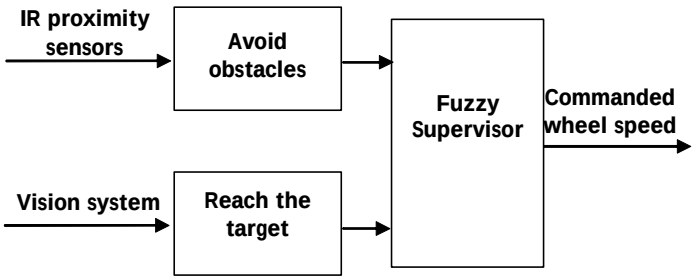


Fig. 2-Behavior based control scheme

3 Implemented control algorithm

In this section we describe the implementation of a control algorithm for the autonomous navigation of a Khepera mobile robot in a 1.2×1.2 m^2 arena with obstacles and a target to be reached. The positions of the target and of the obstacles are not known in advance. As stated before, a web cam is mounted on the top of the arena. Instead than following an internal representation of the path to the target, the navigation algorithm implements a control system based on the reactive paradigm only relying on sensory information. There is a direct link between sensors and actuators in order to by-pass path planning operations that would significantly slow-down the decision process. In our approach, no explicit representation of the world is given to the mobile robot, and the vision system can be considered just as a sensory device measuring the relative position of the robot with respect to the target.

Two simple behaviors, namely *reach the target* and *avoid obstacles*, have been realized with two different fuzzy controllers, henceforth called FLC1 and FLC2 respectively. The *reach the target* behavior only depends on artificial vision information and is the primary task for the robot. The *avoid obstacles* behavior has the highest priority, only depends on IR sensors signals and takes place if an obstacle appears on robot's path. A fuzzy supervisor has been then designed to combine the reference wheel speeds calculated by each behavior layer following a priority code.

The final commanded wheel speeds are then sent to the built-in speed loop of the Khepera. The structure of the designed controller is represented in fig. 2.

The choice of this kind of architecture for our controller has been motivated by the following main advantages that this structure shows with respect to a monolithic solution:

1. debug and tuning operations are faster and easier since each behavior is described by few rules and inputs;
2. the final structure is more flexible as new simple behaviors can be easily added to expand robot skills.

In the following, the design of the two behaviors and of the supervisor are described in detail.

3.1 Reach the target

This behavior reacts to the stimuli of the vision system which provides the information about the relative position between robot and target. The behavior ignores the presence and position of obstacles. The Khepera is equipped with two colored markers on its top for position and orientation detection, and the target is marked with a red spot. The image processing is based on conventional color detections on RGB formats using the Matlab image processing toolbox. The information captured by the vision system are updated every 200 ms and passed through mlib/mtrace to the FLC1 which is running on the dSpace board. According to this behavior, the robot firstly turns until it is aligned to the target, and then moves straight ahead. The distance (DIST) between robot and target and the alignment error (DIR) of the robot are the information provided by the vision system and are the inputs of FLC1. The linguistic labels for fuzzy sets on the DIST input (expressed in millimeters) are *zero*, *near* and *far*, while DIR input is partitioned in *left*, *center-left*, *center*, *center-right*, *right* fuzzy sets. The output variables are the speed command signals for robot wheels whose linguistic labels are *negative fast*, *negative slow*, *zero*, *positive slow*, *positive fast*. We have adopted triangular membership functions (MFs) for the inputs (fig.3) and five uniformly spaced singletons for the output. In table 1, the rule base is reported using AND as conjunction operator. As can be seen, if the alignment error is *zero*, the speed for both wheels will be *positive* and the robot will move straight in the direction of the target; on the other hand if alignment error is *center-left*, then right speed is *negative slow* and left speed is *positive slow* in order to make the robot turn clockwise until the alignment error is *zero*. When DIR is *left* similar considerations hold but the robot turns more rapidly. Finally, when distance is *zero* speed is *zero* for each wheel and the robot stops on the reached target.

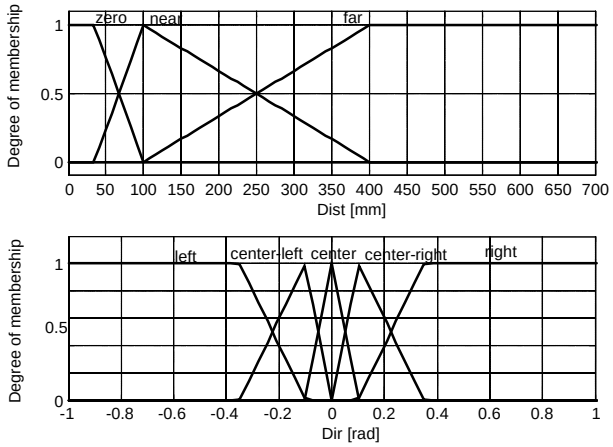


Fig.3 - Input MFs for FLC1

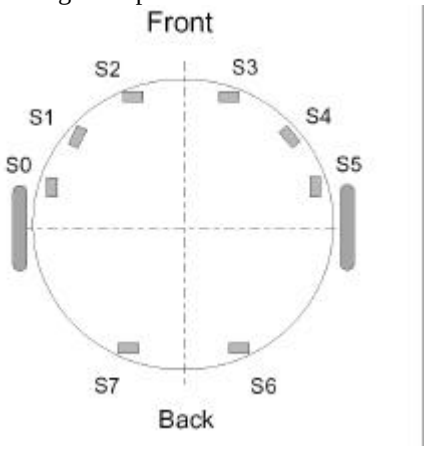


Fig.4 - Input MFs for FLC2

3.2 Avoid obstacles

To guarantee the safe navigation of the Khepera, the FLC2 receives, as inputs, the signals coming from 8 IR proximity sensors and imposes, as outputs, wheel speeds so that robot steers in opposite direction of close obstacles or goes straight ahead in condition of open field. The sensors are labeled after their position on the Khepera from S0 to S7 as reported in fig 4.

We have chosen the same triangular MFs for the eight inputs (see fig.5) in the normalized universe of discourse and five uniformly spaced singletons for the output. The linguistic labels for each input are *far*, *approaching*, *close* and *colliding* depending on the distance between robot and obstacle. In table 2, the rule base representing this behavior is reported using OR as conjunction operator and neglecting for sake of simplicity the rules relative to the back sensors S6 and S7. For example, if an obstacle is on the left (the colliding membership of the S0 sensor is activated), the right wheel speed will be *negative fast* and the left wheel *zero* to make the robot turn clockwise until the sensor no longer detects the obstacle. Instead, according to last rule of the table, if no obstacle is seen by any of the sensors, the robot can proceed straight ahead with the maximum speed.

Table 1- FLC1 Rule table .

INPUTS (logic AND)		OUTPUTS	
Dist	**Dir**	**Right speed**	**Left speed**
zero	*any*	*zero*	*zero*
near	*center*	*positive slow*	*positive slow*
far	*center*	*positive fast*	*positive fast*
any	*left*	*negative fast*	*positive fast*
any	*centre-left*	*negative slow*	*positive slow*
any	*right*	*positive fast*	*negative fast*
any	*centre-right*	*positive slow*	*negative slow*

Table 2 – FLC2 Rule table

INPUTS (logic OR)						OUTPUTS	
S_0	**S_1**	**S_2**	**S_3**	**S_4**	**S_5**	**Right speed**	**Left speed**
coll	*coll*	*coll*	*any*	*any*	*any*	*negative fast*	*zero*
any	*any*	*any*	*coll*	*coll*	*coll*	*zero*	*negative fast*
close	*appr*	*any*	*any*	*any*	*any*	*negative fast*	*positive slow*
any	*any*	*any*	*any*	*appr*	*close*	*positive slow*	*negative fast*
far	*far*	*far*	*far*	*far*	*far*	*positive fast*	*positive fast*

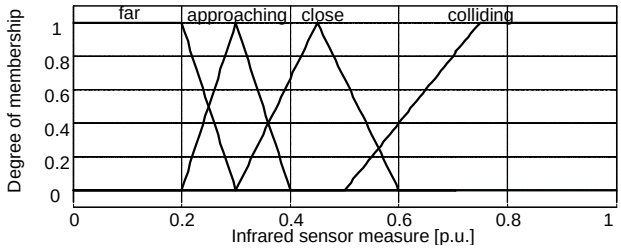


Fig.5 - Input MFs for FLC2

3.3 Fuzzy supervisor

The fuzzy supervisor determines the priority of execution for the two elementary behaviors according to the proximity of obstacles. The maximum value of the signals provided by proximity sensors (henceforth called *max_prox*), and the reference speed for the two wheels, provided by FLC1 and FLC2 respectively, are the input signals of the supervisor, whereas the outputs are the commanded speeds sent to the Khepera.

If *max_prox* is *close*, an obstacle is dangerously close and thus, independently from the actual location of the target, the robot has to avoid the collision. In this case, absolute priority is assigned to the *avoid obstacles* behavior, while *reach the target* is neglected (the outputs of the supervisor coincide with those of FLC2). If *max_prox* is *far*, then robot's path is clear and only the *reach the target* behavior is executed (the outputs of the supervisor coincide with those of FLC1). In every intermediate condition between the two mentioned extremes, the supervisor will perform a fusion of the two FLCs blending their outputs to achieve a safe navigation toward the target.

It must be pointed out that even if many other strategies for switching between these two simple behaviors could be easily conceived, we adopted a hierarchical fuzzy approach envisioning future research developments including the integration of multiple and more advanced behaviors in the

proposed scheme. As stated in [3], in such cases fuzzy supervision offers a transparent and extremely effective solution based on hi-level linguistic decision rules.

4 Experimental results

The navigation experiments have been conducted in a 1.2×1.2 m^2 arena with white obstacles (undetectable to the camera) and a red spot representing the target that the Khepera has to reach. In fig. 6, it is shown a panoramic view of the robot environment. A fine tuning of the fuzzy controllers has been realized monitoring the navigation performances of the Khepera with Control Desk. We want to underline here that these modifications can be performed quite easily, changing the parameters in the Simulink scheme, repeating the experiment and observing the behavior of the robot until good results are obtained. Using high level software such as Matlab/Simulink and Control Desk, this trial and error procedure is greatly simplified.

Fig. 7 shows the navigation performances of the robot. The robot path has been reconstructed using the information provided by the camera and then has been overlaid with the map of the environment (which is not exploited for control purposes). The real proportions among obstacles, target and Khepera have been faithfully reproduced. After some adjustments of the MF positions, the robot is able to successfully navigate in its environment and reach the target, despite of changing light conditions, obstacles and noise. In order to show the performance of the fuzzy supervisor, Fig. 8 a shows the trend of *max_prox* and fig.8 b the right wheel reference speeds provided by the *avoid obstacles* and the *reach the target* behavior and by the fuzzy supervisor. For sake of clarity, even if the whole experiment lasts about 40 seconds, in both figures just the time interval ranging from 4 to 10 seconds is considered. The path of the robot during this period of time is enlightened in grey in fig. 7. Fig. 8a also reports the membership functions of *max_prox*, *close* and *far*, used for the fuzzy supervisor. From 4 to about 6.5 seconds, *max_prox* is *far* and the output of the supervisor coincides with the output of FLC1. The robot is just turning on itself to point toward the target and no obstacles are in its neighborhood. The two peaks in fig. 8 a refer to the two first obstacles the robot finds on its way. The first obstacle is not very close and the fuzzy supervisor blends the output of FLC1 and FLC2 so that the robot just slightly deviates from the straight path to the target. The second obstacle is directly on robot's path and is approached around time instant 9 s. In this case, *max_prox* is *close* and, as can be seen in fig. 8 b, the output of the supervisor is equal to the output of FLC2 in order to avoid the collision.

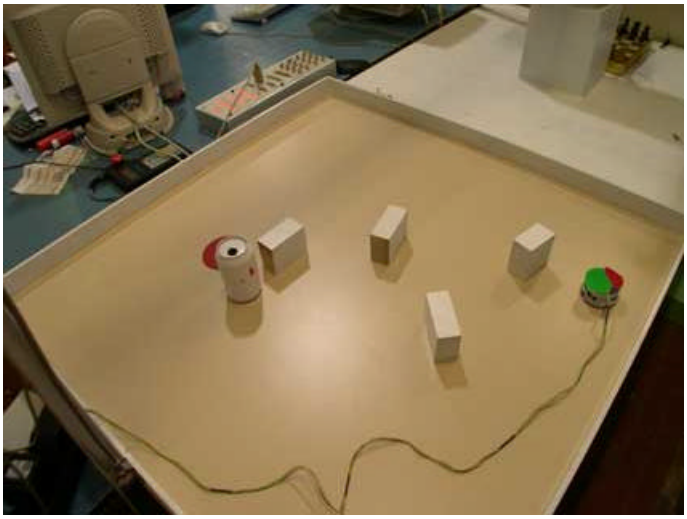


Fig.6 – Panoramic view of the Khepera environment.

5 Conclusions

This paper has described the experimental implementation of a behavior-based fuzzy control system for autonomous navigation of a mobile robot in an unknown and dynamic environment. For the real-time implementation of this control architecture, a rapid prototyping platform has been developed using Matlab/Simulink hi-level programming environment, in order to relieve the designer from coping with tangled hardware and software issues and to permit a straightforward design and an intuitive monitoring of the experiment. The proposed results enhance the effectiveness of the fuzzy-based autonomous navigation strategy and, most importantly, prove the distinctive potentialities of the developed experimental platform in terms of modularity, transparency, and easiness of design in view of more challenging experiments. In fact, in order to increase robot skills to solve difficult tasks in complex scenarios, future research will be devoted to the implementation of further behaviors, to the use of a broader sensor suite (e.g. camera on board, ultrasonic sensors) and to the cooperation among different robots. Thanks to the flexibility of the proposed platform, a noticeable reduction of the development time is expected.

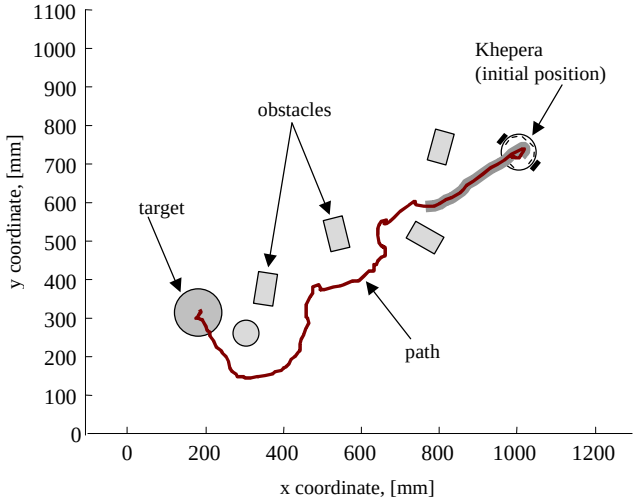
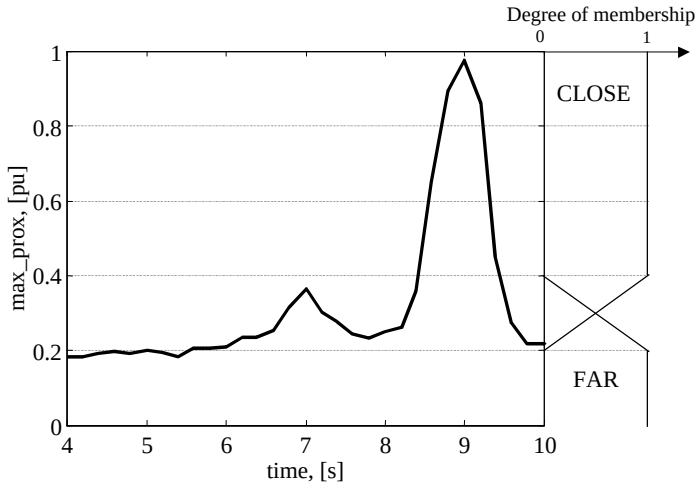
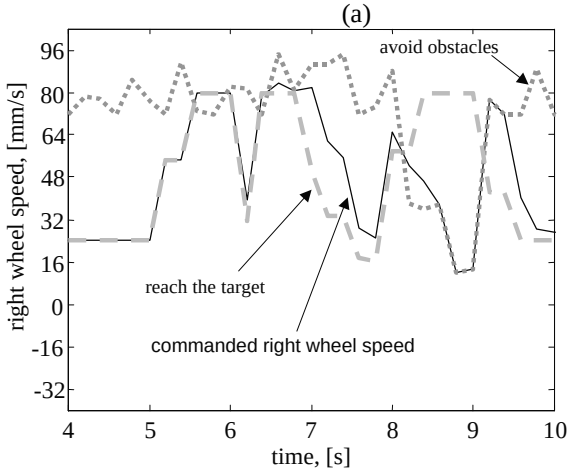


Fig.7 – Measured path of the mobile robot in real world environment



(a)



(b)

Fig.8 - Maximum value of the signals provided by proximity sensors (a) and right wheel reference speeds provided by the *avoid obstacles* and the *reach the target* behavior and by the fuzzy supervisor (b)

References:

[1] Monti A., Santi E., Dougal A. R., Riva M, Rapid prototyping of digital controls for power electronics, *IEEE trans. on Power Electronics*, vol.18., no.3, may 2003.

[2] Brooks R. A., A robust layered control system for a mobile robot, *IEEE Journal of Robotics and Automation*, , vol. RA-2, no.1, march 1986.

[3] Driankov D., Saffiotti A., Fuzzy logic techniques for autonomous vehicle navigation, *Springer Verlag*.

[4] Er M. J., Lim M. T., Lim H. S., Real time hybrid adaptive fuzzy control of a SCARA robot, *Microprocessor and Microsystems 25, pages 369-378, 2001*.

[6] Braitenberg V,.Vehicles, Experiments in synthetic psychology, *The MIT press, 1984.*

[7] Nolfi , Floreano D, Evolutionary robotics, The biology, intelligence and technology of self-organizing machines, *The MIT press, Cambridge Massachusetts.*

[8] Murphy R., Introduction to AI robotics, *The MIT press, Cambridge Massachusetts.*

[9] Chen L., Chiang C., New approach to intelligent control systems with self-exploring process, *IEEE trans. on Systems Man and Cybernetics, part B.*, vol.33, no.1, February 2003.

[10] Ye C., Yung N. H. C., Wang D., A fuzzy controller with supervised learning assisted reinforcement learning algorithm for obstacle avoidance, *IEEE trans on Systems Man and Cybernetics, part B.*, vol.33, no.1, February 2003.

[11] Lee S. I., Cho S. B., Emergent behaviors of a fuzzy sensory-motor controller evolved by genetic algorithms, *IEEE trans. on Systems Man and Cybernetics, part B.*, vol.31, no.6, December 2001.

[12] Serajy H., Howard A., Behavior-based robot navigation on challenging terrain: a fuzzy logic approach, *IEEE trans. on Robotics and Automation*, vol.18 no.3 June 2002.

[13] Rusu P., Petriu E., Whalen T., Cornell A., Spoelder H., Behavior-based neuro-fuzzy controller for mobile robot navigation, *IEEE trans. on Instrumentation and Measurement*, vol.52 no.4 August 2003.

[14] Li H., Yang S. X., A behavior-based mobile robot with a visual landmark-recognition system, *IEE/ASME trans. on Mechatronics*, vol.8 no.3 September 2003.

[15] Khepera II user manual v1.1-K-Team S.A. 2001.