1999 European Control Conference (ECC)
31 August – 3 September 1999, Karlsruhe, Germany

RECENT ALGORITHMS FOR FUZZY AND NEUROFUZZY PATH PLANNING AND NAVIGATION OF AUTONOMOUS MOBILE ROBOTS

Costas S. Tzafestas and Spyros G. Tzafestas

Intelligent Robotics and Automation Laboratory
Electrical and Computer Engineering Department
National Technical University of Athens
Zografou 157 73, Athens, Greece
Fax : +30-1-7722490, e-mail : tzafesta@softlab.ece.ntua.gr

Abstract

This paper reviews a number of recent algorithms for mobile robot path planning, navigation and motion control, which employ fuzzy logic and neuro-fuzzy learning and reasoning. Starting with a discussion of the structure of fuzzy and neuro-fuzzy systems, two fuzzy obstacle avoidance path planning algorithms are presented followed by a 3-level neuro-fuzzy local and global path planning scheme. Then the motion planning and control problem is considered. A fuzzy path tracking strategy is outlined, followed by a fuzzy navigation algorithm among polygonal obstacles and a learning-by-doing neuro-fuzzy motion planning scheme. The paper ends with a hybrid robust motion control technique which combines the minimum interference and sliding mode control principles with fuzzy inference. A representative set of examples are included which illustrate the performance of the algorithms under various realistic conditions.

1 Introduction

The basic robot path planning problem in known environment is the following [1-6]: "*Given a robot and the description of the environment, plan an obstacle collision free path between an initial and a final position and orientation*". It is desirable that the collision-free paths be the shortest among all collision-free paths and have maximum minimum clearance among the paths along the collision free path. The difficulty of the obstacle avoidance problem depends on whether obstacles are moving or not, and on whether the environment and moving objects are known in advance or not. The path planner has to search for optimal paths if the environment is totally known and static, or determine the path of the robot if the trajectories of the moving obstacles are known as well. If the environment is partially or completely unknown the computation has to be carried out on-line under real-time conditions. In both cases the result can be either a set of points to be reached one after another, or a set of directions and velocities. A hybrid method consists in computing first an optimal path off-line and then determining the effective trajectory on-line by initially following the path and eventually avoiding unexpected obstacles [5]. This approach can be followed in well-known environments, such as a warehouse, where some objects (workers, etc.) move around in a non predetermined way. The constraints of on-line or off-line computing induce almost systematically the use of respectively local or global path planning methods.

Global path planning techniques employ a map of the robot's environment off-line together with a variety of approaches to deal with the configuration of position spaces and to determine optimal or suboptimal paths. The principal categories of off-line obstacle avoidance techniques are : *projection, retraction* and *optimal motion planning techniques, configuration space techniques, decomposition* and *distributed representation techniques, gradient* and *potential field techniques, variational programming techniques,* and finally *multi-agent techniques.* If the environment where the robot moves is partially or completely unknown, the global (off-line) techniques cannot be applied. Here local path control techniques for dynamically avoiding the obstacles are needed {see e.g. [3,4]}.

This paper reviews some recent algorithms for fuzzy and neuro-fuzzy path planning and navigation of mobile robots. Due to space limitations the discussions are necessarily limited, but the reader can find detailed results in the references.

2 Structure of fuzzy and neuro-fuzzy systems

Fuzzy logic and fuzzy reasoning was initiated by Zadeh (1965) and permits one to handle vague, imprecise and ill-

defined knowledge in an exact mathematical way. Fuzzy or linguistic algorithms (FLAs) and controllers (FLCs) have found particular applications in robotics [7-27]. Robotic systems are complex systems that cannot be modeled precisely even under specific assumptions and approximations. In many cases the control of robotic systems needs intervention of human operators who employ experiential rules that can cast into the fuzzy logic framework [7-8].
The general structure of any FLA/FLC is shown in Fig. 1 and involves four principal units :

- Fuzzification interface (FI)
- Knowledge-base (KB)
- Fuzzy inference mechanism (FIM)
- Defuzzification interface (DI)

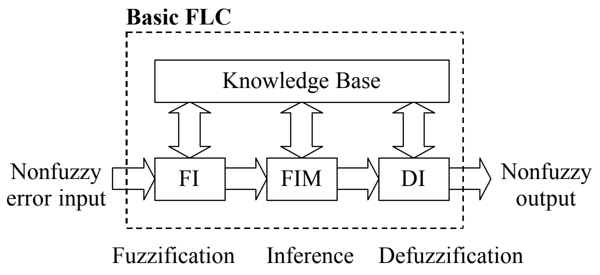


Fig. 1 General structure of FLCs

The fuzzification interface performs the following operations

- Measures the values of inputs
- Maps the input range of values to suitable universes of discourse
- Fuzzifies the incoming data (i.e. converts them to appropriate fuzzy form).

The knowledge base involves a numeric 'data base' section and a fuzzy (linguistic) rule-base section. The data base section involves all numeric information required to specify the fuzzy control rules and fuzzy reasoning. The fuzzy rule base involves the specification of the control goals and control strategies in linguistic form.

The fuzzy inference mechanism, which constitutes the core of the FLC, involves the required decision making logic (fuzzy reasoning such as generalized modus ponens, Zadeh's max-min composition, etc).

The defuzzification interface carries out the following tasks :

- Maps the range of output variables into corresponding universes of discourse
- Defuzzifies the result of FIM (i.e. converts to nonfuzzy form the control action received from FIM).

The fuzzy rule base contains the rules that are to be used for the control of the process. These rules are usually the result of interviews with the expert operators (very rarely come out of mathematical analysis or simulations) and have the IF-THEN form. In the general case the rules have many inputs and many outputs (MIMO). However, it can be shown that a set of MIMO rules can be transformed to a set of MISO (multi input – single output) rules.
Neurofuzzy systems are the product of combining the representation of fuzzy systems with the learning process of neural networks. A neurofuzzy system is initialized and verified using an FLA, while its information processing structure involves three or five – layer neural networks [29]. A typical neurofuzzy system architecture is shown in Fig. 2 which is similar to a lattice-based associative memory neural network structure such as the CMAC and radial basis functions (RBF) networks [30].

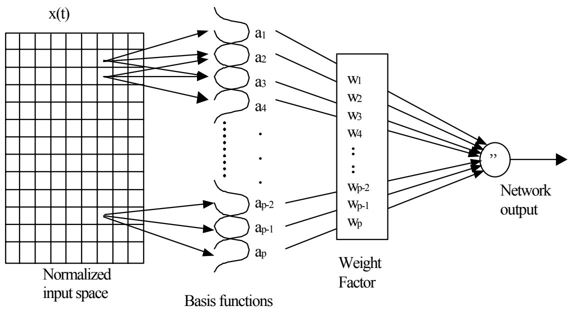


Fig. 2 Typical general architecture of neurofuzzy systems

In Fig. 2, the a_i's (i=1,2,…,p) represent the input basis functions (fuzzy sets) which can be B-spline functions or Gaussian basis functions, etc, that lead to good local generalization and regularization conditions. The lattice-like structure enables simple, well defined network construction, and pruning algorithms to be formulated.

3 Fuzzy and neurofuzzy robot path planning and navigation

3.1 *Fuzzy Path Planning in Mobile Robots*

Representative works in this area include [12, 13, 17, 19, 20, 21]. In [19] the constraints and the solution style adopted are similar to those followed by a human decision maker when trying to solve a path planning problem for a mobile robot. The human never knows precisely the position of the goal, and so he (she) makes use of its direction and the information obtained about the distances, the shape of the obstacles and their visible side, which is strongly fuzzy.
Since there are some rules that affect the output but do not depend on the input data, the system was divided in two subsystems, namely a subsystem which is connected to the input, and a subsystem which is connected to the input but is modified by the output values. Examples of two rules of this type are :

1. If a movement direction has been selected it is not advisable to change it without prior consideration unless some other much better direction is brought to the robot's attention.

2. If the robot returns to a point that has passed before, it should ? an alternative route (direction) in order to avoid endless loops and so reach a dead-end.

The fuzzy path planning algorithm of the robot (represented by a point in the space) which has to move from an initial position S_0 (starting point) to a final position G (goal) works as follows. We first define a coordinate frame attached to the robot system with Ox-axis the straight line SiG, and divide the space into K directions with reference the SiG (or Ox) direction. Obviously two consecutive directions have an angular distance of 360/K degrees. It is advised to select the number K of directions such that the axes Ox, Ox′, Oy and Oy′ are involved in them. Now, from the sensors' measurements and the fuzzy inference on the data base we determine and associate to each direction a *priority*. Then, we select the direction with the highest priority and the new point S_{i+1} is placed in this direction with (S_iS_{i+1})=□, where □ is the step length that we select. The steps of the proposed path planning algorithm are the following :

Step 1: Obtain the measurements from the position S_i

Step 2: Fuzzify the results of these measurements

Step 3: Introduce the fuzzy results into the rules adopted

Step 4: Determine a priority value for each direction

Step 5: Multiply the above priority value by a given *a priori* weight for each direction

Step 6: Determine the direction with the highest priority and move by a step length in this direction to find S_{i+1}

Step 7: Repeat the algorithm for the position S_{i+1}.

In [34], reinforcement learning is used to help a car-like mobile robot to avoid obstacles and go to a target with no *a priori* knowledge. The learning-to-drive algorithm consists of two terms; a "critic" transforming the environmental feedback to a higher quality heuristic signal, and an "actor" actually controlling the robot, which learns exploiting the signal provided by the critic. The Sugeno type fuzzy reasoning is used to implement the actor and the critic. Reinforcement is an intelligent kind of learning algorithm that tries to establish an input/output mapping which maximizes a scalar function (called reinforcement) of the environment feedback. The core of the algorithm is a neurofuzzy architecture named ASAFES 2 (Adaptive Stochastic Algorithm for Fuzzy computing/function EStimation) [31] based on the Sugeno fuzzy reasoning method [32, 33].

3.2 *Neurofuzzy Mobile Robot Navigation*

Here a complete mobile robot-navigation system consisting of three cooperating subsystems [22, 23] will be reviewed. The first subsystem performs *local* navigation (obstacle avoidance and goal seeking), the second performs *global* navigation, and the third performs *map recalling*. The first subsystem is based on an actor-critic-type reinforcement learning neurofuzzy controller, the second employs a topologically ordered Hopfield neural network, and the third subsystem is actually a Hopfield neural network, with a non-local learning rule, used as an associative memory. The core of the system is the second (global navigation) subsystem which uses a certainty grid-type representation of the environment, called *eligibility factor matrix*, which is a map built on the fly utilizing ultrasonic sensor information, taking into account the inaccuracy of the sensors, and odometry for mobile robot positioning. The system was tested by simulation.

The *local navigation module* is a reinforcement learning neurofuzzy controller (driving the robot motors) which is designed via the ASAFES 2 system discussed above. The output value is a weighted average of the output values of each activated fuzzy rule, and is applied directly to the motors of the robot. The reinforcement signal is available after each iteration taking values in the interval [-2, +2] where the value –2 corresponds to robot failure and the value +2 to robot success (reaching the target within a distance of 10 length units). To enhance the behavior of the algorithm the membership functions were allowed to be more fuzzy during the first stages of learning, and a momentum term with a high learning rate was employed in the weight updating scheme. Also the gradient descent method was used for parameter identification of the input variable membership functions.

The *global navigation module* is a Hopfield neural network with analog neurons, topologically ordered to represent the environment. At every time step this module is run and provides finally a virtual target placed near the robot. The environment is represented on a lattice-like network. The environment map is constructed on the way using the ultrasonic sensor headings, under the assumption that the position and orientation of the mobile robot can be calculated from dead reckoning. Since the sensor signals are not accurate the idea of the *eligibility factor matrix* was used, allowing the elements of the environment matrix to take real values in [0,1].

The eligibility factor matrix has some similarities to the certainty grid and the vector histogram. The updating is performed so as the eligibility factor of elements corresponding to free space (obstacle area) is increased (decreased) by an amount depending on the distance from the robot.

Finally, the *map recalling module* is an ordinary Hopfield network used as an associative memory which tries to recall and complete a partially observed environment. This network has 64×38 neurons, with a one-to-one correspondence to the neurons of the global navigation module. For learning, a non-local learning rule was used [34] which exhibits a very high capacity of N patterns, where N is the number of neurons.

The state value of the neuron corresponding to the target position is set to 1, while the states of the neurons corresponding to obstacles are clamped to value 0. At every time step the Hopfield network is initialized (all neuron states are set to 0 except for the target neuron which is set to 1).

This three-level navigation system can be used as part of integrated intelligent service mobile robots that can follow commands like "bring food to nest", "push box", "follow wall", or "open door".

4 Mobile robot motion planning and control

In [15-18] fuzzy path tracking strategies were applied to a four-wheel vehicle (RAM-1), where the two parallel wheels are driven by d.c. motors and the front and rear wheels are steered by a d.c. motor with kinematic rigid link. The locomotion system can provide a zero turning radius. For path tracking the front and rear wheels are steered and the parallel wheels are used with differential drive. The fuzzy path tracking algorithm has the structure of Fig. 1. Consider a goal point in the path at a lookahead distance away as shown in Fig. 3(a). The goal point has a negative x coordinate (negative dx) and the path tangent at this point (path heading) is the same with the vehicle's heading (d□=0). Then, the vehicle must be turned to the left by a certain amount as shown in Fig. 3(a). Similarly, if dx=0 but d□>0 the vehicle has to turn again to the left by a certain amount (Fig. 3(b)). The actual fuzzy control rules have the form :

IF dx is NL AND d□ is Z AND *l* is L THEN $\square_r$ is PL

where $\square_r$ is the steering command (new curvature), *l* is the distance between the current position of the robot and the desired point on the map (lookahead distance), Z = near zero, L = large, and NL(PL) = negative (positive) large.

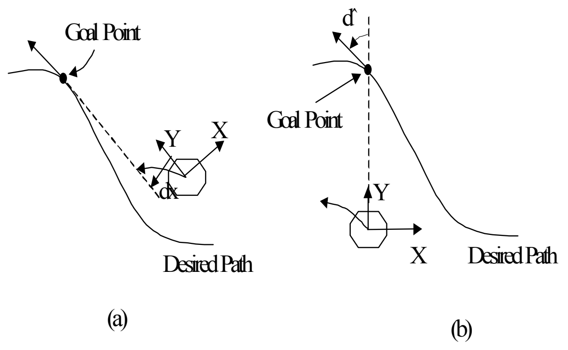


Fig. 3 Illustration of the fuzzy path tracking strategy :
(a) negative dx,d□=0 (turn left), (b) dx=0, d□>0 (turn left)

To cope with actual navigation conditions, the above basic FLC was enhanced by including a fuzzy supervisory level in order to be able to perform real-time automatic tuning of the controller parameters.

In [25] a sensor-based fuzzy algorithm is developed for navigating a mobile robot in a 2-dimensional unknown environment with stationary polygonal obstacles. In this algorithm, initially, the robot scans and measures the angles and the distances of all visible vertices of the obstacles by the sensors. The priority of each visible vertex is determined by feeding the readings of the sensors to a suitable set of priority fuzzy rules.

Table 2 shows an instance of assigning priorities, where of course the choice of priority assignments is not unique. One can assign priorities to Table 2 that depend on his(her) experience or knowledge about the system.

When the priorities of all visible vertices are found, the navigation algorithm selects the highest one and guides the robot to the corresponding vertex. The navigation process is repeated until the target point g is reached. The determination of the ranges of the fuzzy variables in Fig. 7 influences strongly the performance of the fuzzy navigation algorithm which involves the following steps.

Step 1. The initial (temporary) starting point s and the goal point g are given.

Step 2. If the point g is visible from the point s, which means that a navigation path has been found, terminate the iteration process and go to Step 9; otherwise do:

Step 3. Scan and measure the angles and the distances of all visible vertices of the obstacles.

Step 4. Using the fuzzy rules given in Table 2, determine the priorities of all visible vertices.

Step 5. Select the vertex □ with the highest priority among the visible vertices that have not been visited before

Step 6. Navigate the robot from point s to vertex □ along the line segment $\overline{s\upsilon}$.

Step 7. Select the vertex □ as the new temporary starting point s.

Step 8. Go to step 2.

Step 9. End.

In [26] a neurofuzzy motion planning technique is developed for semi-autonomous mobile robots.

Table 2 : Fuzzy rules assigning priorities of visible vertices

		Distance (*)			
		Very Near	Near	Far	Very Far
Angle	Very Small	7	6	5	4
	Small	6	5	4	3
	Large	5	4	3	2
	Very Large	4	3	2	1

(*) Quantized number 7 represents the highest priority and the quantized number 1 denotes the lowest priority.

Fuzzy representations are employed for the robot to learn a repertory of behaviors from an instructor which is then refined using neural learning-by-doing techniques.The motion planning algorithm developed is applied to an idealized robot (called MITOS) moving in an unknown terrain. MITOS can steer left, right, straight or in any direction in between and its speed can be controlled, and also is capable of functioning in a changing environment where the position of the objects are not known, motor control is dynamically exerted, and optimization is of secondary concern. This paper includes a good review of neurons built via fuzzy tools (fuzzy neurons). A case study is provided which concerns the motion planning of MITOS in dynamic environments (where the path the robot travels has walls on both sides, it is on a level floor and has a limited number of intersections as it happens in buildings with long corridors and intersections).
The robot learns from a human automobile driver while operating a car. A total number of 27 fuzzy rules were instructed to the robot which provide a limited but representative repertory for collision avoidance. The robot accepts as inputs the environmental data from the robot's sensors. The two action (control) variables are y_1 (steering) and y_2 (speed). The value of y_1 defines the direction of MITOS motion (left, right or in any direction in between). The value of y_2 defines the speed of the robot while it is making a turn. The data objects defined as fuzzy variables are : *road width* (x_1), *obstructing object size* (x_2), *obstructing object position* (x_3), *steering* (y_1), *speed control* (y_2). A rule example is : "IF x_1 is WIDE AND x_2 is SMALL AND x_3 is LEFT THEN y_1 is MEDIUM-SPEED". The max-min composition formula is used by MITOS to evaluate its fuzzy rules. All combinations of the fuzzy values are matched against the 27 rules in the rule-base of the instructed behavior.
We close our review with the robust fuzzy motion control method of [27]. This method combines the sliding mode control principle with fuzzy logic. The resulting motion controller which assures stability is called *Reduced Complexity Sliding Mode Fuzzy Logic Controller* (RC-SMFLC) and possesses robustness and simplicity. The dynamic model of the mobile robot is

$$\left(\boldsymbol{m} + \boldsymbol{I_R}(\boldsymbol{s})\dot{\boldsymbol{v}}(\boldsymbol{s})\right) + \boldsymbol{I'_R}(\boldsymbol{s})\boldsymbol{v}^2(\boldsymbol{s}) + \boldsymbol{mgz'}(\boldsymbol{s}) = \boldsymbol{u}(\boldsymbol{s})$$
$$\textbf{\textit{where}}\ \mathrm{v}(\mathrm{s}) = {\boldsymbol{ds}}/{\boldsymbol{dt}},\ (\cdot)' = {\boldsymbol{d}(\cdot)}/{\boldsymbol{ds}},\ \boldsymbol{I_R}(\boldsymbol{s}) \le \Omega(\boldsymbol{s}), \Omega(\boldsymbol{s}) > \boldsymbol{I}$$
$$\boldsymbol{I} = \text{robot's moment of inertia}$$

around the center of gravity G, m=robot's mass, g=gravity constant, z(s) is the height of the center of gravity with respect to a world coordinate frame, □=angular speed, and u(t)=vector of actuator signals. The actuator constraints are : $-F_2 \le u_i \le F_1$ $(i=1,2,\ldots,n_w)$ where $-F_2$ and F_1 are the brake and accelerating bounds of the actuators, respectively, and n_w is the total number of actuators. The steering force is constrained as : $-U_2 \le u \le U_1$, where $U_1 = cn_wF_1$ and $U_2 = cn_wF_2$ with c depending on the range of the steering angles of the wheels and the maximum inclination of the floor. Collision avoidance is ensured through the so-called *Minimum Interference Strategy* (MIS) formulated in [28]. The set-points of the velocity for which collision with moving obstacles is avoided are determined in advance. The robust controller designed in the paper ensures that these set-points are reached without being affected by the uncertainty of the robot environment or the parametric uncertainties in the robot's model.
To design now the sliding-mode FLC two possible control actions are assumed : *increase* or *decrease* the control signal U, and the following rule base is considered :

$$\mathrm{R}_1 : \mathrm{IF}\ \mathrm{sgn}(\mathrm{e}(\mathrm{t})\dot{\mathrm{e}}(\mathrm{t})) < 0\ \mathrm{AND}\ \ f\mathrm{u}_\mathrm{k} > 0\ \mathrm{THEN}\ \ f\mathrm{u}_{\mathrm{k}+1} > 0$$
$$\mathrm{R}_2 : \mathrm{IF}\ \mathrm{sgn}(\mathrm{e}(\mathrm{t})\dot{\mathrm{e}}(\mathrm{t})) < 0\ \mathrm{AND}\ \ f\mathrm{u}_\mathrm{k} < 0\ \mathrm{THEN}\ \ f\mathrm{u}_{\mathrm{k}+1} < 0$$
$$\mathrm{R}_3 : \mathrm{IF}\ \mathrm{sgn}(\mathrm{e}(\mathrm{t})\dot{\mathrm{e}}(\mathrm{t})) > 0\ \mathrm{AND}\ \ f\mathrm{u}_\mathrm{k} > 0\ \mathrm{THEN}\ \ f\mathrm{u}_{\mathrm{k}+1} < 0$$
$$\mathrm{R}_4 : \mathrm{IF}\ \mathrm{sgn}(\mathrm{e}(\mathrm{t})\dot{\mathrm{e}}(\mathrm{t})) > 0\ \mathrm{AND}\ \ f\mathrm{u}_\mathrm{k} > 0\ \mathrm{THEN}\ \ f\mathrm{u}_{\mathrm{k}+1} > 0$$

where □u_k is the change in the control signal at the kth iteration of the algorithm.
To achieve convergence, the nonlinear transfer characteristic of the fuzzy controller should be such that the smaller the distance from the setpoint is, the smaller the change of the control signal becomes. The control law in RC-SMFLC is similar to the sliding-mode one, the only difference being in the term K(x,t)sat(s/□) which now becomes $\mathrm{K}[\mathrm{sgn}(\mathrm{e_k e_{k+1}})]\mathrm{sgn}(\mathrm{e_k \dot{e}_k})$. That is, we have

$$\boldsymbol{u} = \hat{\boldsymbol{b}}^{-1}(\tilde{\boldsymbol{u}} - \hat{\boldsymbol{f}}),\ \tilde{\boldsymbol{u}} = \boldsymbol{G}\left[\hat{\boldsymbol{u}} - \boldsymbol{K}\,\mathrm{sgn}(\boldsymbol{e_k e_{k+1}})\mathrm{sgn}(\boldsymbol{e_k \dot{e}_k})\right]$$
$$\hat{\boldsymbol{u}} = \boldsymbol{x}_{\boldsymbol{d}}^{(\boldsymbol{n})} - \sum_{\boldsymbol{k}=1}^{\boldsymbol{n}-1}\binom{\boldsymbol{n}-1}{\boldsymbol{k}}\lambda^{\boldsymbol{k}}\boldsymbol{e}^{(\boldsymbol{n}-\boldsymbol{k})}$$

5 Some illustrative examples

Example 1 : Mobile-Robot Obstacle Avoidance
The direction selection technique of section 3.2 [19] is applied. The distance D from the point S_i to some obstacle is measured by a sensor, quantized in eleven (11) subintervals and fuzzified as shown in Table 3. The entries in this table are the values of the membership functions □$_j$(D) in the various subintervals distributed among the linguistic values L = Low, VL = Very Low, ML = More or Less Low, M = Medium, NM = Not Medium, H = High, MH = More or Less High, UN = Unknown, etc. The knowledge is represented by fuzzy matrices where Zadeh's max-min inference rule is used. The fuzzification of an input "a" is to set □$_A$(x)=1 when x=a and □$_A$(x)=0 when x≠a. Defuzzification is carried out using the center of gravity method. The number of possible directions of movement is selected equal to K=16 and the step length equal to b=5 pixels. The rules employed are :

R_1 : IF D_i is L THEN P_i is H
R_2 : IF D_i is M THEN P_i is M
R_3 : IF D_i is H THEN P_i is L

where P_i is the priority of the ith direction quantized and fuzzified as shown in Table 3. To avoid turning around the obstacles (which increases the travelling time of the robot) three rules concerning the difference of the distance from the

Table 3 : Distance quantization and fuzzification

	Range	L	VL	ML	NL	ME	MM	NM	HI	VH	MH	NH
0	D<15	1.00	1.00	1.00	0.00	0.00	0.00	1.00	0.00	0.00	0.00	1.00
1	15<D<60	0.67	0.45	0.82	0.33	0.00	0.00	1.00	0.00	0.00	0.00	1.00
2	60<D<105	0.33	0.11	0.57	0.67	0.25	0.50	0.75	0.00	0.00	0.00	1.00
3	105<D<150	0.00	0.00	0.00	1.00	0.50	0.71	0.50	0.00	0.00	0.00	1.00
4	150<D<195	0.00	0.00	0.00	1.00	0.75	0.87	0.25	0.00	0.00	0.00	1.00
5	195<D<240	0.00	0.00	0.00	1.00	1.00	1.00	0.00	0.00	0.00	0.00	1.00
6	240<D<285	0.00	0.00	0.00	1.00	0.75	0.87	0.25	0.20	0.04	0.45	0.80
7	285<D<330	0.00	0.00	0.00	1.00	0.50	0.71	0.50	0.40	0.16	0.63	0.60
8	330<D<375	0.00	0.00	0.00	1.00	0.25	0.50	0.75	0.60	0.36	0.77	0.40
9	375<D<420	0.00	0.00	0.00	1.00	0.00	0.00	1.00	0.80	0.64	0.89	0.20
10	D>420	0.00	0.00	0.00	1.00	0.00	0.00	1.00	1.00	1.00	1.00	0.00

obstacles that are encountered in the neighboring directions were added in the rule base :

R_4 : IF DD_i is L THEN P_i is L
R_5 : IF DD_i is M THEN P_i is M
R_6 : IF DD_i is H THEN P_i is H

where DD_i is the maximum of the difference of the distance from the obstacles that are met in the two neighboring directions. Assuming that these differences are measured, the results obtained are much better.

Example 2 : Reinforcement learning-based local path planning
Here the reinforcement technique of [34] (see sec. 3.1) was applied to a car-like robot (10 units long, 3 units wide) with maximum steering angle 45° and control variables the *steering angle* and *direction of motion* (forward, backward). Five ultrasonic sensors were used for forward motion that cover the left, front-left, front, front-right and right areas. The maximum distance reading for the front looking sensors is 125 length units and less than 20 units for the lateral sensors. Minimum reading for all sensors is 7 length units. The range of each sensor reading is divided in two fuzzy sets : NEAR and FAR with logistic membership function. Since both the actor and critic methods have the same input partitioning, two networks of 3×2×2×2 = 24 neurons each are used. Four input variables are used *: left, front, right sensor readings*, and *heading error*. To keep the number of tunable weights low only the left and right sensor readings are involved in the consequence part of the fuzzy rule, i.e.

$$y = w_0 + w_1 \cdot \text{input}_L + w_2 \cdot \text{input}_R$$

where input_L (input_R) is the left (right) input distance. The environment used is a constrained rectangle of 600 units × 600 units area. The robot is assumed to have reached the target if the distance from it is smaller than 10 length units. The target, after being reached, appears on the other side of the environment, causing the robot to go through the obstacles to find it. A learning session was considered successful if the robot could reach the target for more than 100 times, and in the meanwhile did not collide with obstacles for more than 100 times. With a speed of 4 length units per time step in more than 600 learning sessions all except one were successful with 40.5 collisions on the average.

Example 3 : Mobile Robot Path Tracking
The path tracking method of [15-18] was applied to the RAM-1 robot as described in Section 4. The robot was called to track a reference path at a speed of 1.4 m/s starting from a distance 1.8m away (□x = −1.8) with zero error in orientation and curvature (see Fig. 4 for the definition of variables).

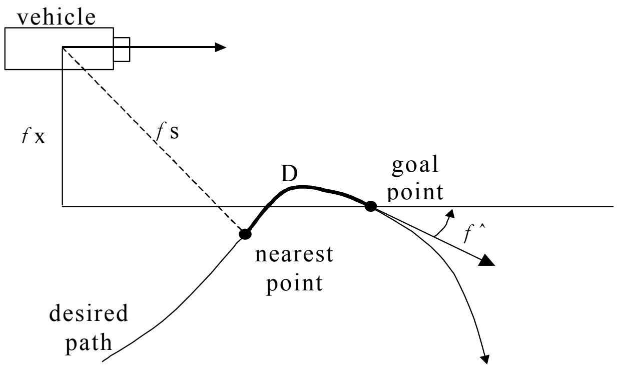
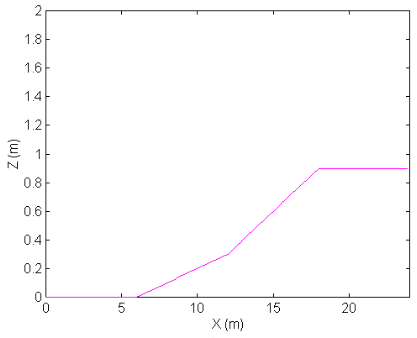


Fig. 4 Set-up and definition variables

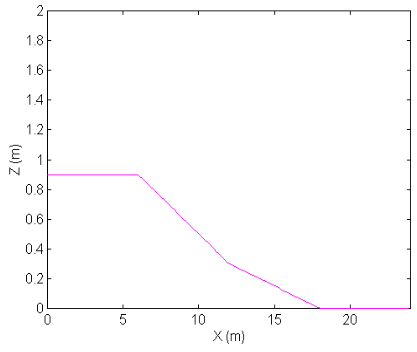
The path followed by the robot and the corresponding curvature depend on the lookahead distance. A long lookahead distance leads to a smooth convergence to the desired path, but the performance of tracking is not good from the meeting point onwards. The tracking improves if a short lookahead distance is used but the transition is more abrupt (oscillatory), although the vehicle reaches the desired path more quickly. This situation is corrected by using an enhanced (supervisory) fuzzy path tracking controller (for details see [26,29]).

Example 4 : Robust Neurofuzzy Motion Control
The performance of the hybrid sliding-mode fuzzy controller of [27] (sec. 4) was tested in several cases of robot motion in a partially known environment. Uncertainty was considered to exist only in the slopes $\partial z/\partial s$ of the robot trajectory. The robot

could mount an uphill slope ($\partial z/\partial s > 0$) or could go down a downhill slope ($\partial z/\partial s < 0$). Both the magnitude and the signum of the slope were unknown and time varying. The uphill slope was initially chosen to be 5% and afterwards was further increased to 10%. The desirable velocity was 4.2m/s. Then the robot descends a downhill slope. The human action in each case is to press/depress the gas pedal or the break pedal until the desired velocity is reached. Thus the sliding-mode FLC has to act both as an accelerator or as a break in the robot motion. The downhill slope was initially selected to be –10% and afterwards was set to –5%. In both cases the performance of the hybrid SMFLC controller was excellent, with very fast convergence of the velocity to the desired setpoint and limited oscillations.
The velocity fluctuations and the robot trajectory in uphill and downhill motions are shown in Fig. 5.



(a)



(b)

Fig. 5. Controller performance in uphill (a) and downhill (b) motion

6 Conclusions

The field of autonomous path planning, navigation and motion control of mobile robots for obstacle avoidance is at a very mature state and many techniques, such as penalty function, distance function, path velocity decomposition, cell decomposition, potential field, and vector force field methods are by now classical. The class of fuzzy and neuro-fuzzy techniques for path/motion planning is about 10 years old and constitutes a challenging area of research. From the survey of the small but representative set of techniques provided in this paper it follows with certainty that the results obtained are very promising. These techniques do not need accurate measurements (sensors' headings) or skilled *a priori* experiential knowledge. Therefore they seem to be more appropriate for handling uncertain and unknown environments with various randomly moving objects and obstacles. Using simple sensing techniques in conjunction with fuzzy logic reduce the processing time up to real-time levels. Fuzzy descriptions can be used to acquire desirable behaviors from an instructor and subsequently refine them in a learning-by-doing fashion using adaptive neural learning techniques.

REFERENCES

1. R. Brooks, Solving the find-path problem by good representation fo the free space, *IEEE Trans. Syst. Man Cybern.,* Vol. 13, No. 3 , pp. 190-197, 1983.
2. L. Gouzenez, Strategies for solving collision-free trajectories problems for mobile and manipulator robots, *The Intl. J. of Robotics Research,* Vol. 3, No. 4, pp. 51-65, 1984.
3. O. Khatib, Real-time obstacle avoidance for manipulators and mobile robots, *The Intl. J. of Robotics Research,* Vol. 5, No. 1, pp. 90-98, 1986.
4. B.H. Krogh and C.E. Thorpe, Integrated path planning and dynamic steering control for autonomous vehicles, *Proc. IEEE Intl. Conf. on Robotics and Automation,* pp. 1664-1669, 1986.
5. J.C. Latombe, Robot Motion Planning, *Kluwer Academic Publishers*, Boston, 1991.
6. P.C.-Y. Sheu and Q. Xue, Intelligent Robotic Planning Systems, *World Scientific*, Singapore, 1993.
7. S.G. Tzafestas and A.N. Venetsanopoulos, Fuzzy Reasoning in Information, Decsion and Control Systems, *Kluwer Academic Publishers*, Dordrecht/Boston, 1994.
8. S.G. Tzafestas and C.S. Tzafestas, Fuzzy and Neural Intelligent Control : Basic Principles and Architectures, In : S.G. Tzafestas (ed.), *Methods and Applications of Intelligent Control, Kluwer Academic Publishers*, Dordrecht/Boston, 1997.
9. B.A.M. Wakileh and K.F. Gill, Use of fuzzy logic in robotics, *Computers in Industry*, Vol. 10, pp. 35-446, 1988.
10. R. Palm, Fuzzy controller for a sensor guided robot manipulator, *Fuzzy Sets and Systems*, Vol. 31, No. 2, pp. 133-149, 1989.
11. K. Althoefer, P. Zavlangas and B. Krekelberg, Adaptive fuzzy navigation for robotic manipulators, *Proc. EUFIT'97 - The 5th European Congress on Intelligent Techniques and Soft Computing*, Aachen, Germany, 1997.
12. J. Gasos, M.C. Garcia-Allegre and R. Rosa, Fuzzy local navigation of a simulated real robot in unknown

environment, *Proc. IEEE Intl. Workshop on Intelligent Motion Control*, pp. 445-449, 1990.

13. S. Ishikawa, A method of indoor mobile robot navigation by using fuzzy control, *Proc. IROS'91 : Intl. Workshop on Intelligent Robots and Systems*, pp. 1013-1018, 1991.
14. C. Kemal, Fuzzy rule based motion controller for an autonomous mobile robot, *Robotica*, Vol. 7, No. 1, pp. 37-42, 1989.
15. J.L. Martinez, A. Ollero and A. Garcia-Cerezo, Fuzzy strategies for path tracking of autonomous vehicles, *Proc. EUFIT'93 : The 1st European Congress on Fuzzy and Intelligent Technologies*, Aachen, pp. 24-30, 1993.
16. M. Maeda, Y. Maeda and S. Mukarami, Fuzzy drive control of an autonomous mobile robot, *Fuzzy Sets and Systems*, Vol. 39, pp. 195-204, 1991.
17. H. Maaref and C. Barret, Fuzzy help in mobile robot navigation, *Proc. IEEE Intl. Conf. on Robotics and Automation*, Taipei, Taiwan, pp. 865-871, May 1995.
18. A. Garcia-Cerazo, J.L. Martinez and A. Ollero, Fuzzy direct path tracking of explicit paths in mobile robots : Application to RAM-1, *Proc. EURISCON'94 : The 2nd European Robotics, Intelligent Systems and Control Conference*, Malaga, Spain, pp. 253-260, 1994.
19. S.G. Tzafestas and G.B. Stamou, A fuzzy path planning algorithm for autonomous robots moving in an unknown and uncertain environment, *Proc. EURISCON'94 : The 2nd European Robotics, Intelligent Systems and Control Conference*, Malaga, Spain, pp. 140-149, 1994.
20. S.G. Tzafestas and K.C. Zikides, A mobile robot guidance system based on three neural network modules, *Proc. Second ECPD Conf. on Advanced Robotics, Intelligent Automation and Active Systems*, Vienna, Austria, 1996.
21. S.G. Tzafestas, K.C. Zikides and G.B Stamou, A reinforcement learning fuzzy logic controller for the mobile robot local path planning problem, *Proc. Intl. Symp. on Soft Computing*, Reading, U.K., 1996.
22. S.G. Tzafestas and K.C. Zikides, Complete mobile-robot navigation via neural and neurofuzzy control, *Proc. 1st Mobinet Symp. : Mobile Robotics, Technology for Health Care Services*, Athens, Greece, pp. 205-218, May 1997.
23. S.G. Tzafestas and K.C. Zikides, A 3-level neuro-fuzzy autonomous robot navigation system, *Proc. ECC'97 : European Control Conference*, Brussels, 1997.
24. J. Gas□s and A. Martin, Mobile robot localization using fuzzy maps, *Proc. IJCAI'95 Workshop on Fuzzy Logic in Artificial Intelligence : Towards Intelligent Systems*, Montreal, Canada, pp. 207-224, Aug. 1995.
25. C.-J. Wu, A learning fuzzy algorithm for motion planning of mobile robots, *J. Intell. & Robotic Systems*, Vol. 11, pp. 209-221, 1995.
26. L.H. Tsoukalas, E.N. Houstis and G.V. Jones, Neurofuzzy motion planners for intelligent robots, *J. Intell. & Robotic Systems*, Vol. 19, pp. 339-356, 1997.
27. G.G. Rigatos, C.S. Tzafestas and S.G. Tzafestas, Robust fuzzy-logic velocity control of a mobile robot for moving obstacles avoidance, Proc. 2nd IMACS Intl. Conf. on Circuits, Systems and Computers (CSC'98), Vol. 1, pp. 41-48, 1998.
28. K.J. Kyriakopoulos and G.N. Saridis, Optimal and suboptimal motion planning for collision avoidance of mobile robots in non-stationary environments, *J. Intell. & Robotic Systems*, Vol. 11, No. 3, pp. 223-267, 1995.
29. C.T. Lin, Neural Fuzzy Control Systems with Structure and Parameter Learning, *World Scientific*, Singapore, 1994.
30. C.J. Harris, M. Brown, K.M. Bossley, D.J. Mills and F. Ming, Advances in neurofuzzy algorithms for real-time modeling and control, *Engineering Applications of Artificial Intelligence*, Vol. 9, No. 1, pp. 1-16, 1996.
31. K.C. Zikides and A.V. Vasilakos, ASAFES 2 : A novel neuro-fuzzy architecture for fuzzy computing based on functional reasoning, *Fuzzy Sets and Systems*, Vol. 83, pp. 63-84, 1996.
32. M. Sugeno and K. Murakami, Fuzzy parking control of a model car, *Proc. 23rd IEEE Conf. on Decision and Control*, Las Vegas, U.S.A., pp. 902-903, 1984.
33. T. Takagi and M. Sugeno, Fuzzy identification of systems and its application to modeling and control, *IEEE Trans. Syst., Man, Cybern.*, Vol. 15, pp. 116-132, 1985.
34. I. Kanter and H. Sompolinsky, Associative recall of memory without errors, *Physical Review A*, Vol. 35, No. 1, pp. 90-98, 1986.
35. S.G. Tzafestas and E.S. Tzafestas, Learning, Reasoning, and Problem Solving in Robotics, In : *S.Y. Nof* (ed.), *Handbook of Industrial Robotics* (2nd edition), John Wiley & Sons, Inc. New York, 1999.
36. S.G. Tzafestas (Guest ed.), Autonomous Mobile Robots in Health Care Services, *J. Intell. & Robotic Systems*, (Special Issue) Vol. 22, Nos. 3-4, 1998.