Fuzzy Logic Control of Robot Manipulator in the Presence of Fixed Obstacle

Salah Kermiche, Saidi Mohamed Larbi, and Hadj Ahmed Abbassi
Automatic and Signal Laboratory, Faculty of Engineering, Annaba University, Algeria

Abstract: *This paper presents a solution for the problem of learning and controlling a 2R-plan robot manipulator in the presence of fixed obstacle. The objective is to move the arm from an initial position (source) to a final position (target) without collision. Potential field methods are rapidly gaining popularity in obstacle avoidance applications for mobile robots and manipulators. The idea of imaginary forces acting on a robot has been suggested by Andrews and Hogen [1983] and Khatib [1985]. Thus, we propose an approach based on potential fields principle, we define the target as an attractive pole (given as a vector directly calculated from the target position) and the obstacle as a repulsive pole (a vector derived by using fuzzy logic techniques). The linguistic rules, the linguistic variables and the membership functions are the parameters to be determined for the fuzzy controller conception. A learning method based on gradient descent for the self tuning of these parameters is introduced. Thus, it is necessary to have an expert person for moving the arm manually. During this operation of teaching, the arm moves and memorizes the data (inputs and outputs). This operation is used to find the controller parameters in order to reach the desired outputs for given inputs.*

Keywords: *Robot manipulator, fuzzy logic control, obstacle avoidance method, self tuning, surface, deep structure.*

Received July 13,2005; accepted March 16, 2006

1. Introduction

Robotics is a relatively new field of modern technology that crosses traditional engineering boundaries. Understanding the complexity of robots and their applications requires knowledge of electrical engineering, mechanical engineering, industrial engineering, computer science, and mathematics. New disciplines of engineering, such as manufacturing engineering, applications engineering, and knowledge engineering, are beginning to emerge in order to solve more complex problem.

Yesterday's teleoperator movements were quite easily controllable by a human operator, but in nowadays the accuracy and complexities of positions of robots may be better achieved by supplementing human capabilities with computer power in order to generate these complex trajectories and to control the robot manipulator, respectively.

Robot manipulator is designed to perform efficiently very complex tasks in cluttered environments. In particular, they are required to move in the presence of fixed or even mobile obstacles, tracking a prescribed path without any collision. Some methods for generating collision-free paths are adapted from mobile robots [2, 6, 10, 11, 12]. Robot serial manipulator needs to avoid both the end-effector and the links. For this reason, their accessible workspace is rather limited unless the number of joints increases.

The manipulator moves in a field of forces where the goal position is an attractive pole and where obstacles and kinematic joint limits are repulsive forces [6]. These two forces determine the arm's orientation; the attractive force is calculated from the goal position and for the repulsive force a fuzzy technique is used. The arm, the obstacle and the target (goal) can take any position inside the workspace. The fuzzy controller using the obstacle avoidance is able to evaluate the repulsive force corresponding to the obstacle's relative position. The learning method allows the self adjustment of the parameters. During manual training, the controller memorises the data. The following method uses an adjustable fuzzy controller for the parameters determination (the number of memberships functions, the linguistic variables, the rules etc..).

2. Modelling

The modelling represents the arm behaviour by algebraic equations, here, a geometric model is used.
The parameters of the arm model are joints and operational positions. The first parameters allows to modify its geometry and the second determines the position and the orientation of the end-effector [5, 7, 8, 9, 14].

The direct geometric model is described by the following equations as shown in Figure 1.

$$X_2 = L_1 Cosq_1 + L_2 Cos\,(q_1 + q_2) \quad (1)$$

$$Y_2 = L_1 Sinq_1 + L_2 Sin\,(q_1 + q_2) \quad (2)$$

And the Inverse Geometric Model (IGM) is described by:

$$q_2 = \pm ArcCos\ \{[X_2^2 + Y_2^2 - (L_1^2 + L_2^2)]/2L_1L_2\} \quad (3)$$

$$q_1 = Arctg\left(\frac{(Y_2/X_2)([L_1 + L_2Cosq_2] - X_2L_2Sinq_2)}{([L_1 + L_2Cosq_2] + Y_2L_2Sinq_2)}\right) \quad (4)$$

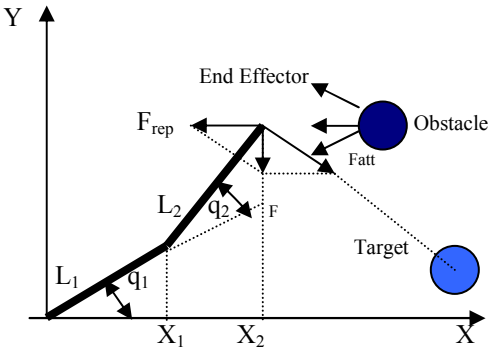


Figure 1. Coordinate frames for two-link planar robot.

3. Collision Avoidance Strategy

A fuzzy system is a system based on the concepts of approximate reasoning: Linguistic variables, fuzzy propositions and linguistic if-then rules. The goal is to realise a fuzzy controller that is able to evaluate the repulsive force (vector) V_{rep} characterising the actual relative position of the obstacle [1, 6].

The controller has two inputs and one output as shown in Figure 2, the inputs are the observation angle θ_{obs} and the distance d_{obs} towards the obstacle, the output is the repulsive vector V_{rep}. The orientation angle depending on V_{or} is the input of the arm and its outputs are the coordinates (x_a, y_a) and the direction θ_a.

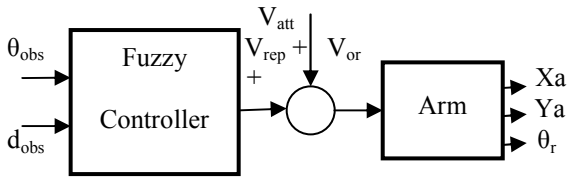


Figure 2. Controller + Arm.

3.1. Fuzzification

Fuzzy sets are used to quantify the information in the rule-base, and the inference mechanism operates on fuzzy sets to produce fuzzy sets; hence, we must specify how the fuzzy system will convert its numeric inputs into fuzzy sets so that they can be used by the fuzzy system.

The fuzzification module performs two tasks:

- Input normalisation, mapping of input values into normalised universes of discourse.
- Transformation of the crisp process state values into fuzzy sets, in order to make them compatible with the antecedent parts of the linguistic rules that will be applied in the fuzzy inference engine.

3.1.1. Fuzzification of the Angle θ_{obs}

We suppose that the arm can perceive an obstacle in a direction inside the interval [-90° 90°]. The membership function is represented by seven fuzzy subsets of Gaussian form, as shown in Figure 3.

LL:L arge Left; ML: Middle Left; SL: Small Left
EZ: Zero Environment; SR: Small Right; MR: Middle Right, LR: Large Right.

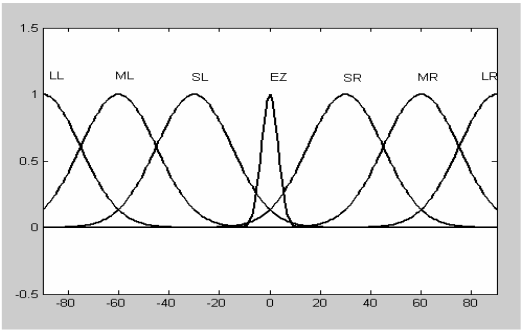


Figure 3. Membership function for the first input variable θ_{obs}.

3.1.2. Fuzzification of the Distance d_{obs}

We admit that the arm can detect an obstacle from a distance of 30 units. The membership function is expressed by three fuzzy subsets as shown in Figure 4.
S: Short; M: Medium; L: Long.

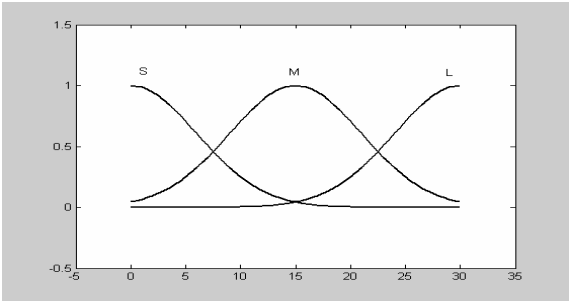


Figure 4. Membership functions for second input variable.

3.1.3. Fuzzification of the Repulsive Angle θrep

The membership function of the repulsive angle has a constant form belonging to the interval [-135° 135], as shown in Figure 5.

3.2. Inference

The inference mechanism has two basic tasks:

- Determining the extent to which each rule is relevant to the current situation as characterized by the inputs u_i, i= 1, 2, …, n (we call this task matching).
- Drawing conclusions using the current inputs u_i and the information in the rule-base (we call this task an inference step).

For matching, note that $A_1^j x A_2^k x \ldots x A_n^l$ is the fuzzy set representing the premise of the i^{th} rule (j, k, …, l; p, q)$_i$. There are then two basic steps to matching:

1. Matching involves finding fuzzy sets with membership functions.
2. We form membership values μ_i $(u_1, u_2, \ldots, u_n)$ for the i^{th} rule's premise that represent the certainty that each rule premise holds for the given inputs.

Let $x_1, x_2, ..., x_m$ be linguistic variables on the input space $X = x_1 * x_2 * \ldots * x_m$, and y be a linguistic variable (or a real variable) on the output space Y; then two forms of fuzzy inference rules by the fuzzy " If ... Then ..." rule model can be described as follows [13]:

Form (1): Fuzzy inference rules by Product-Sum-Gravity Fuzzy Reasoning Method. The fuzzy inference rules are defined as:

Rule 1: If x_1 is A_{11} and x_2 is A_{21} and ... and x_m is A_{m1}
Then y is B_1 (5)
Rule 2: If x_1 is A_{12} and x_2 is A_{22} and ... and x_m is A_{m2}
Then y is B_2 (6)
...
Rule n: If x_1 is A_{1n} and x_2 is A_{2n} and ... and x_m is A_{mn}
Then y is B_n (7)

Where A_{ji} (j = 1, 2, ..., m; i = 1, 2, ..., n) and B_i are fuzzy subsets of x_j and y, respectively and the subscript i corresponds to the ith fuzzy rule.

Form (2): Fuzzy inference rules by Simplified Fuzzy Reasoning Method. The fuzzy inference rules are defined as:

Rule 1: If x_1 is A_{11} and x_2 is A_{21} and ... and x_m is A_{m1}
Then y is y_1 (8)
Rule 2: If x_1 is A_{12} and x_2 is A_{22} and ... and x_m is A_{m2}
Then y is y_2 (9)
...
Rule n: If x_1 is A_{1n} and x_2 is A_{2n} and ... and x_m is A_{mn}
Then y is y_n (10)

Where A_{ji} (j = 1, 2, ..., m; i = 1, 2, ..., n) is a fuzzy subsets of X_j and y_i is a real number on Y .

For example, the representations of these rules would then be constructed as follows:

If (θ_{ods} is LL and d_{ods} is S) then θ_{rep} is APP or

If (θ_{ods} is LL and d_{ods} is M) then θ_{rep} is TPP or

....

If (θ_{ods} is LR and d_{ods} is L) then θ_{rep} is EZ.

The rules are summarized in Table 1.

4. Deffuzzification

A number of defuzzification strategies exist, and it is not hard to invent more. Each provides a means to choose a single output (which we denote with y_q^{crisp}) based on either the implied fuzzy sets or the overall implied fuzzy set (depending on the type of inference strategy chosen).

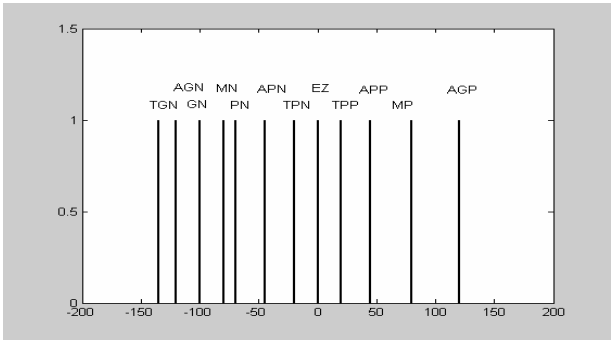


Figure 5. Membership function of the repulsive angle θrep.

Table 1. Fuzzy rule table.

θ_{rep}		LL	ML	θ_{obs} SL	EZ	SR	MR	LR
d_{obs}	S	APP	MP	AGP	TGN	AGN	MN	APN
	M	TPP	APP	MP	GN	MN	APN	TPN
	L	EZ	TPP	APP	PN	APN	TPN	EZ

4.1. Defuzzification: Implied Fuzzy Sets

As they are more common, we first specify typical defuzzification techniques for the implied fuzzy sets:

- *Center Of Gravity (COG)*: A crisp output y_q^{crisp} is chosen using the center of area and area of each implied fuzzy set, and is given by:

$$y_q^{crisp} = \frac{\sum_{i=1}^{R} b_i^q \int_{y_q} \mu_{\hat{B}_q^i}(y_q)\,dy_q}{\sum_{i=1}^{R} \int_{y_q} \mu_{\hat{B}_q^i}(y_q)\,dy_q} \quad (11)$$

Where R is the number of rules, b_i^q is the center of area of the membership function of B_q^p associated with implied fuzzy set $\hat{B}_q^i$ for the i^{th} rule (j, k, …, l, p, q)$_i$, and

$$\int_{y_q} \mu_{\hat{B}_q^i}(y_q)\,dy_q$$

denotes the area under $\mu_{\hat{B}_q^i}(y_q)$.

- *Center-Average*: A crisp output y_q^{crisp} is chosen using the centers of each of the output membership functions and the maximum certainty of each of the conclusion represented with the implied fuzzy sets, and is given by:

$$y_q^{crisp} = \frac{\sum_{i=1}^{R} b_i^q \sup_{y_q}\left\{\mu_{\hat{B}_q^i}(y_q)\right\}}{\sum_{i=1}^{R} \sup_{y_q}\left\{\mu_{\hat{B}_q^i}(y_q)\right\}} \quad (12)$$

Where *sup* denotes the *supremum* (i. e., the least upper bound which can often be thought of as the maximum value).

4.2. Takagi-Sugeno Fuzzy Systems

For the functional fuzzy system, we use singleton fuzzification, and the i^{th} MISO rule has the form:

$$\begin{aligned} &\text{if } \widetilde{u}_1 \text{ is } \widetilde{A}_1^j \text{ and } \widetilde{u}_2 \text{ is } \widetilde{A}_2^k \text{ and}, \ldots, \\ &\text{and } \widetilde{u}_n \text{ is } \widetilde{A}_n^l \text{ then } b_i = g(\cdot) \end{aligned} \tag{13}$$

Where (.) simply represents the argument of the function g_i and the b_i are not output membership function centers. The premise of this rule is defined the same as for the MISO rule for the standard fuzzy system in equation (8). The consequents of the rules are different, however. Instead of a linguistic term with an associated membership function, in the consequent we use a function $b_i = g_i(.)$ (hence the name "functional fuzzy system") that does not have an associated membership function. Notice that often the argument of g_i contains the terms $u_i, \quad i = 1, 2, \ldots, n,$ but other variables may also be used.

For the functional fuzzy system, we can use an appropriate operation for representing the premise, and defuzzification may be obtained using:

$$y = \frac{\sum_{i=1}^{R} b_i \mu_i}{\sum_{i=1}^{R} \mu_i} \tag{14}$$

$$\mu_i(u_1, u_2, \ldots, u_n) = \mu_{A_1^i}(u_1) * \mu_{A_2^k}(u_2) * \ldots * \mu_{A_n^l}(u_n)$$

It is interesting to notice that for this type of defuzzification, we do not need to define the widths of the membership functions. It follows that a set of output membership functions can be defined as illustrared in Figure 5. This type of membership functions is called singletons. This definition corresponds to the special case of Takagi and Sugeno's controller.

5. Adjustable Fuzzy Controller

Tuning fuzzy rule-based systems for linguistic fuzzy modelling is an interesting and widely developed task. It involves adjusting some of the components of the knowledge base without completely redefining it. This contribution introduces a method based on gradient descent for jointly fitting the fuzzy rule symbolic representations and the meaning of the involved membership functions. This task is usually developed by means of linguistic fuzzy rule-based systems, which use fuzzy rules composed of linguistic variables taking values in a term set with a real-world meaning. Thus, the fuzzy linguistic model consists of a set of linguistic description regarding the behaviour of the system being modelled.

Each of these linguistic fuzzy rules may be represented at two different levels of description by defining two different structures:

- *Surface Structure*: It is a less specific description and involves defining the rule in its symbolic form as relation between input and output linguistic variables.
- *Deep Structure*: It is a more specified description and consists of the surface structure together with the definitions of the membership functions associated to the linguistic terms of the variables.

5.1. Linguistic Rules

We suppose that the controller has M + K linguistic variables, M inputs, K outputs and N linguistic rules (8, 9, 10).

5.2. Learning Method

In this section, it is used a supervised learning method for the adaptation of parameters of the Takagi and Sugeno's controller. The learning task consists of modifying the controller's parameters in order to obtain the desired output for given inputs. Adaptation of parameters is achieved by presenting pairs of input and desired output vectors to the system and applying an adaptive algorithm which adjusts the parameters, by minimising a measure of the error between the desired output and the actual output of the system. The scheme of this procedure is shown in Figure 6.

5.3. General Algorithm

The parametric model is given in the following form [4]:

$$\begin{aligned} \frac{du}{dt} &= f(u, x, z, t) \\ y &= g(u, x, z, t) \end{aligned} \tag{15}$$

All solutions to parameter estimation problems consist of finding the extremum of criterion function V considered as a function of the parameters of the unknown system. The criterion function is usually given as:

$$V = E\{(\vec{e}(t))^2\} \tag{16}$$

$$\text{Or} \quad V = E\left\{\frac{1}{N}\sum_{t=1}^{N}(\vec{e}(t))^2\right\} \tag{17}$$

E: Means.
N: Number of iterations.
e (t): Learning error vector.

In order to minimise the learning error, we will find the minimum of the criterion function V in equation (12). It can be found by solving:

$$-\nabla_z V = \left[-\frac{\partial \nabla}{\partial z_1}, \ldots\ldots\ldots, \frac{-\partial \nabla}{\partial z_p} \right] = 0 \tag{18}$$

$-\nabla_z V$: Is the notation for the gradient of V.

P: Number of adjustable parameters.

Robbins and Monro suggested the following scheme to solve equation (14) recursively as time goes on:

$$z_p(t+1) = z_p(t) - \Gamma_p \nabla_z V\left[z_p(t)\right] \tag{19}$$

Γ_p: Is the predefined constant named learning rate.

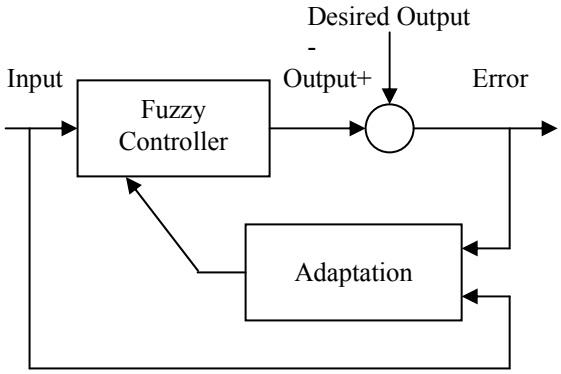


Figure 6. A supervised learning scheme.

5.4. Adaptation of Parameters of Takagi and Sugeno's Controller

The Takagi and Sugeno's fuzzy controller has three types of parameters to adapt:

- Centre values $a = (a_{11},\ldots,a_{nm},\ldots,a_{NM})^T$.
- Width values $b = (b_{11}, \ldots, b_{nm}, b_{NM})^T$.
- Consequences values

$$c = (c_{11},\ldots,c_{nk},\ldots,c_{NK})^T,$$

It follows that vector in equation (15) is

$$\vec{Z} = (a_{11},\ldots,a_{NM},b_{11},\ldots,b_{NM},c_{11},\ldots,c_{NK})^T \tag{20}$$

The number of parameters to adapt is:

$$P = 2N \times M + K \times N$$

The vector which minimizes the criterion function is given by:

$$\frac{-\partial V}{\partial a_{11}},.,\frac{-\partial V}{\partial a_{NM}},\frac{-\partial V}{\partial b_{11}},.,\frac{-\partial V}{\partial b_{NM}},\frac{-\partial V}{\partial c_{11}},.,\frac{-\partial V}{\partial c_{NK}} = 0$$

And the recursive (learning) rules:

$$a_{nm}(t+1) = a_{nm}(t) - \Gamma_a \frac{\partial V(z)}{\partial a_{nm}} \tag{21}$$

$$b_{nm}(t+1) = b_{nm}(t) - \Gamma_b \frac{\partial V(z)}{\partial b_{nm}} \tag{22}$$

$$c_{nk}(t+1) = c_{nk}(t) - \Gamma_c \frac{\partial V(z)}{\partial c_{nk}} \tag{23}$$

The membership functions of the controller are Gaussians, then the partial derivatives of the criterion V are:

$$\frac{\partial V}{\partial a_{nm}} = \sum_{k=1}^{K}(y_k - y_{dk}) \frac{u_n}{\sum_{n=1}^{N} u_n}(c_{nk} - y_k)\frac{x_m - a_{nm}}{b_{nm}^2}$$

$$\frac{\partial V}{\partial b_{nm}} = \sum_{k=1}^{K}(y_k - y_{dk}) \frac{u_n}{\sum_{n=1}^{N} u_n}(c_{nk} - y_k)\frac{(x_m - a_{nm})^2}{b_{nm}^3}$$

$$\frac{\partial V}{\partial c_{nk}} = (y_k - y_{dk}) \frac{u_n}{\sum_{n=1}^{N} u_n}$$

By substituting $\frac{\partial V}{\partial a_{nm}}$, $\frac{\partial V}{\partial b_{nm}}$ *and* $\frac{\partial V}{\partial c_{nk}}$ into equations (17, 18, 19), the adaptation of the parameters of the Gaussians and weights is done by:

$$a_{nm}(t+1) = a_{nm}(t) - \Gamma_a \frac{u_n}{\sum_{n=1}^{N} u_n} \frac{x_m(t) - a_{nm}(t)}{b_{nm}(t)^2}$$

$$\sum_{k=1}^{K}(y_k(t) - y_{dk}(t))(c_{nk}(t) - y_k(t)),$$

$$b_{nm}(t+1) = b_{nm}(t) - \Gamma_b \frac{u_n}{\sum_{n=1}^{N} u_n} \frac{(x_m(t) - a_{nm}(t))^2}{b_{nm}(t)^3}$$

$$\sum_{k=1}^{K}(y_k(t) - y_{dk}(t))(c_{nk}(t) - y_k(t));$$

$$c_{nk}(t+1) = c_{nk}(t) - \Gamma_c \frac{u_n}{\sum_{n=1}^{N} u_n}(y_k(t) - y_{dk}(t))$$

5.5. Linguistic Rules Extraction

The problem of the linguistic rules extraction is to convert the parameters (a_{nm}, b_{nm}, c_{nk}) at the end of the adjustment to linguistic values. To solve this problem, we compare the membership functions defined by a_{nm} and b_{nm} with preset ones whose linguistic terms belong to a set of trajectories provides by an expert person.

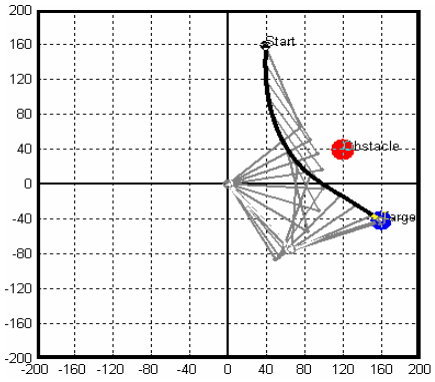
The controller parameters are identified by an off-line procedure based on the memorised data and the initial parameters.

- Number of inputs M = 2.
- Number of outputs K = 1.
- Number of rules N = 21.

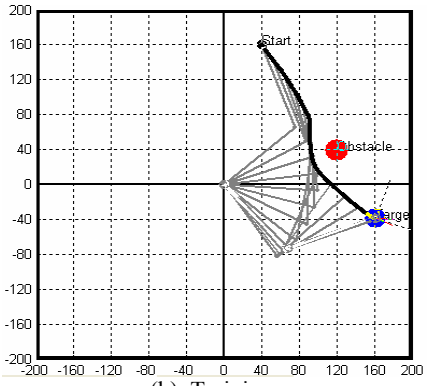
- Learning rate = $\Gamma_a = 0.05$, $\Gamma_b = 0.05$, $\Gamma_c = 0.1$.

6. Simulation

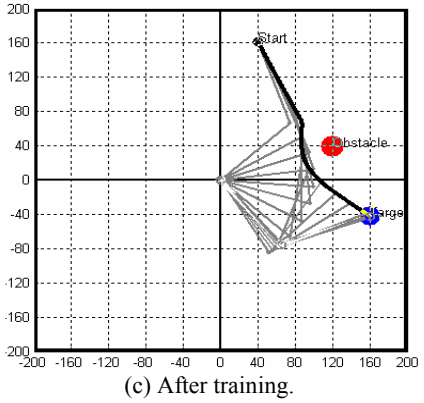
The source, the target and the obstacle positions are specified. Before training, the arm moves from a start configuration to a goal configuration without collision as shown in Figure 7-a. Figure 7-b shows the trajectory specified by the operator from the initial position to the final position (training). Figure 7-c describes the arm trajectory with collision avoidance after training.
We note that the trajectory after training is optimal compared to the one before training as shown in Figure 7-a. We have observed that after training, the concentration of singletons is located on left side. This is due to the avoidance of obstacle which is done on the right side Figures 7-d, 7-e, 7-f.



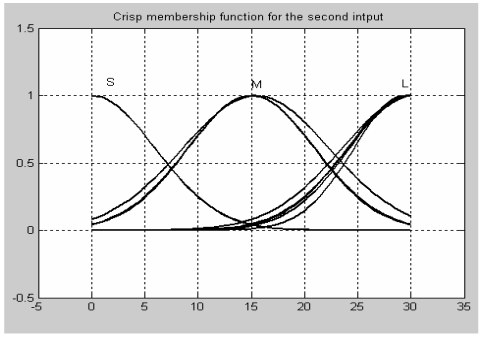
(a) Before training.



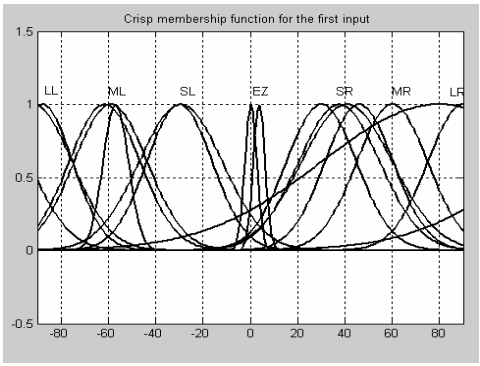
(b) Training.



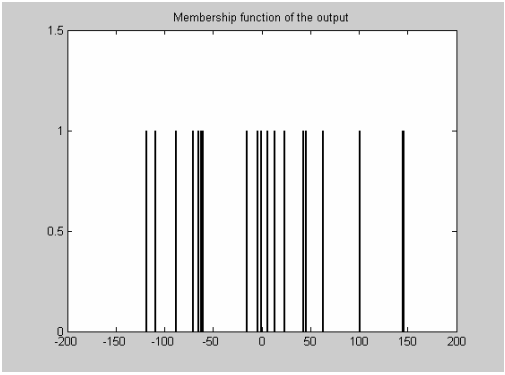
(c) After training.



(d) Crisp membership functions for the second input.



(e) Crisp membership functions for the first input.



(f) Crisp membership functions for the output.

Figure 7. Simulation example.

7. Conclusion

In this paper, we have presented a solution to the problem of trajectory tracking without collision. The arm motion depends on the forces field approach. The manipulator moves in a field of forces where the goal position is an attractive pole and where the obstacle is a repulsive pole. The attractive force is calculated from the goal position and the repulsive force is determined by a fuzzy logic. The manipulator has to follow a trajectory specified by the operator from a start configuration to a goal configuration, which goes through a fixed obstacle. When a potential collision with the obstacle is detected, the collision avoidance redirects the arm motion to the repulsive force in order to generate a new collision free path. The method has been tested on two-link robot arm and the results are very satisfactory.

References

[1] Buhler H., "Réglage Par Logique Floue," *Technical Report*, Presses Polytechniques et Universitaires Romandes, 1994.

[2] Faverjon B., "Obstacle Avoidance Using Octree in the Configuration Space of a Manipulator," *in Proceedings of the IEEE International Conference on Robotics and Automation*, Atlanta, USA, March 1987.

[3] Foulloy L. and Galichet S, *Typology of Fuzzy Controllers: A Comparaison of Fuzzy and Linear Controller*, John Wiley and Sons, New York, 1995.

[4] Godjevac J., "Neuro-Fuzzy Controllers, Design, and Application," *Technical Report*, French UniversityP olytechnic Presses, 1997.

[5] Kermiche S. and Abbassi H. A., "Commande d'un Bras Manipulateur Par Logique Floue," *in Proceedings of the SNAS 2002*, Annaba, Algeria, 2002.

[6] Khatib O., "Real Time Obstacle Avoidance for Manipulators and Mobile Robots," *The International Journal of Robotics Research*, vol. 5, no. 1, pp. 90-98, 1986.

[7] Krzysztof K., *Modelling and Identification in Robots*, Springer-Verlag, London Limited, 1998.

[8] Lallemand J. P. and Zeghloul S., *Robotique Aspects fondamentaux*, Masson, 1994.

[9] Li W., Tanaka K., and Wang H. O., "Acrobatic Control of a Pendubot," *IEEE Transactions on Fuzzy Systems*, vol. 12, pp. 549-559, 2004.

[10] Lozano-Perez T., "Spatial Planning: A Configuration Space Approch," *IEEE Transactions on Computers*, vol. C-32, no. 2, pp. 108-120, 1983.

[11] O'Dunlaing C. and Yap C. K., "A Retraction Method for Planning the Motion of a Disk," *Journal of Algorithms*, vol. 6, no. 1, pp. 104-111, 1985.

[12] Shi Y. and Mizumoto M., "Some Considerations on Conventional Neuro-Fuzzy Learning Algorithms by Gradient Descent Method," *Elsevier Fuzzy Sets Systems*, vol. 112, 2000.

[13] Sharir M. and Schorr A., "On Shortest Paths in Polyhedral Spaces," *in Proceedings of the 16th ACM Symposium on Theory of Computing*, Washington DC, 1984.

[14] Spong M. W. and Vidyasagar M., *Robot Dynamics and Control*, John Wiley and Sons, 1989.

Salah Kermiche received his BEng and MPhil degrees in control engineering, from Badji Mokhtar University, Algeria, in 1985 and 1989, respectively. Currently, he is a senior lecturer in the Department of Electronic Engineering, Badji Mohktar University, Annaba, Algeria. His research interests include intelligent control, modelling, and robotics.

Saidi Mohamed Larbi received his BEng and MPhil degrees in control engineering, from Badji Mokhtar University, Algeria, in 1990 and 1997, respectively. Currently, he is a senior lecturer in the Department of Electronic Engineering, Badji Mohktar University, Annaba, Algeria. His research interests include general predictive control, neural networks, and robotics.

Hadj Ahmed Abbassi received his BEng degree in control engineering from Badji Mokhtar University, Algeria, in 1985 and his PhD degree in control engineering from Reading University, UK, in 1991. Since 2003, he has been appointed as a professor in the Department of Electronic Engineering, Badji Mohktar University, Algeria. His research interests include general predictive control, intelligent control, modelling, identification, fault detection, diagnosis, and robotics.