Adaptive Two Layer Fuzzy Control of a Mobile Robot System.

by

M. Mohammadian*, R J Stonier**
*Department of Computer Science
Edith Cowan University, Perth Western Australia
m.mohammadian@cowan.edu.au
**Department of Mathematics and Computing
Central Queensland University
Rockhampton Qld Australia
r.stonier@cqu.edu.au

Abstract

Various approaches have been proposed for solving collision avoidance problems in robotic systems [1, 2, 3, 4, 5, 6]. In this paper we propose a method for designing an adaptive two layer fuzzy control system for collision-free trajectory control of a multiple robot system. The proposed method is based on the integration of Genetic Algorithms and Fuzzy Logic. The robots are considered as point masses moving in a common work space. The knowledge base which controls how to navigate in the work space without collision is determined by learning via Genetic Algorithms.

1. Introduction

Fuzzy Logic Control (FLC) is an active research area in modern control [7, 8, 9, 10]. It has been found useful when the process is either difficult to control or difficult to model by conventional methods. Although its application to industrial processes has often produced results superior to classical control [7, 8, 11], the design procedures are limited by the heuristic rules of the system. It is assumed that a significant process change does not occur that is outside the fuzzy knowledge based system. This implicit assumption limits the application of a fuzzy logic controller to the case of normal working conditions for which the fuzzy knowledge based system is capable of handling. To accommodate abnormal working conditions adaptive functions can be introduced to adjust the parameters of a fuzzy control system to meet any unexpected case that may arise. Some self-organising fuzzy logic controllers have been reported, cf [1, 8, 10].

In this paper genetic algorithms are used as a self-organising mechanism to learn fuzzy rules of a two layer fuzzy logic control system. This integrated architecture is applied to the problem of determining collsion-free trajectory control of two mobile robots in the plane.

2. Fuzzy Logic Control (FLC)

Several approaches to fuzzy logic control development are possible. In general a compositional "rule of inference", a mathematical statement describing how the linguistic variables are to be manipulated, is employed to control the problem environment. The first step is to determine which variables will be important in choosing an effective control action. Any number of these decision variables may appear, but the more that are used, the larger the rule set that must be written. Decision variables are simply the variables upon which the control decision is based. It is known [13], that the total number of rules in a system is an exponential function of the number of system variables. For a multi dimensional system, it may become unproductive to realise the fuzzy rule-based controller. To overcome this problem, the authors in [14] use a hierarchical fuzzy control structure in which the most influential parameters are chosen as the system variables in the first level, the next most important parameters are chosen as the system variables in the second level. In the hierarchy, the first level gives an approximate output which is then modified by the second level rule set. This procedure is repeated in succeeding levels of the hierarchy. The number of rules in a complete rule set is reduced to a linear function of n

0-7803-2759-4/95/$4.00

by the hierarchy, instead of an exponential function of n [13]. A few methods for transforming human knowledge or experience into the rule base and database of a fuzzy inference system have been suggested, cf [1, 15]. These methods try to generate the rule-base of a system by learning all parameters of the system at once. Their success is only limited, and if there is a need to add or remove one of the parameters of the rule-base, the whole rule-base must be rebuilt. In this paper, we design an adaptive two layer FLC system for the control of two mobile robots considered as point masses moving in a plane. A genetic algorithm (GA) [16], is used as an adaptive learning method to define a new layer of fuzzy rules which can be incorporated easily into an existing FLC knowledge base. Based on the rule-base generated by the FLC performance a second fuzzy rule base is constructed.

3. Collision Avoidance application using FLC and GA

We use a two layer Adaptive Fuzzy Logic Control (AFLC) system. For our robot application the first level rule set controls the basic direction of a robot from an initial position, and the second level rule set controls each robots individual speed. A block diagram model for the control of the robot system is given in Figure 1 illustrating its major parts: the AFLC system, GA system and plant (robot system).

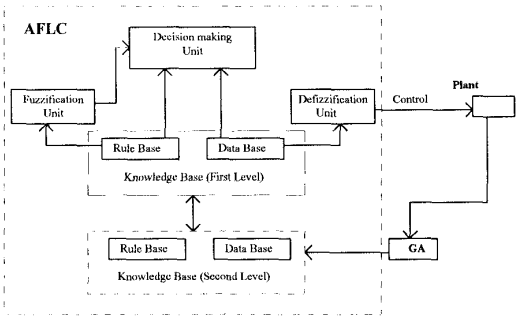


Figure 1. AFLC and GA model

We assume that there exists a separate fuzzy knowledge base to drive each robot to its target using constant speed with heading angle as control; for example see [1, 15]. A GA is then used to build a new fuzzy knowledge base that extends this existing knowledge and incorporates fuzzy rules to control *both* direction and speed. There is a mechanism built into the second layer to control each robot's speed in order to avoid collision. We give below a description of the knowledge base in layer one.

The fuzzy rules have two inputs x,ϕ and single-output $\bar{y}$. Here x, represents the x-position (horizontal coordinate) and ϕ the heading angle measured relative to the horizontal, respectively. The output $\bar{y}$ is the steering angle Θ It is assumed that there exists enough clearance between robot and the target so we can ignore the y-position (vertical coordinate) of the robot, for details see [8, 15]. We divide the domain intervals into appropriate regions, and assign a fuzzy membership function, see Figure 2.

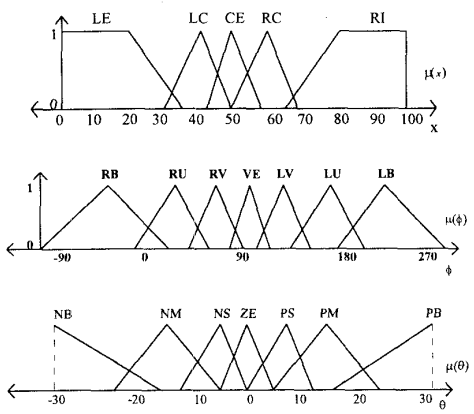


Figure 2 Fuzzy regions and the corresponding membership functions of x, ϕ and Θ

Consider now the second layer of fuzzy rules that controls the speed of each robot. The analysis is performed by taking each robot in turn, that is, robot one moves first then robot two, then robot one etc. Consider without loss of generality the control of robot one, similar construction follows for robot two.

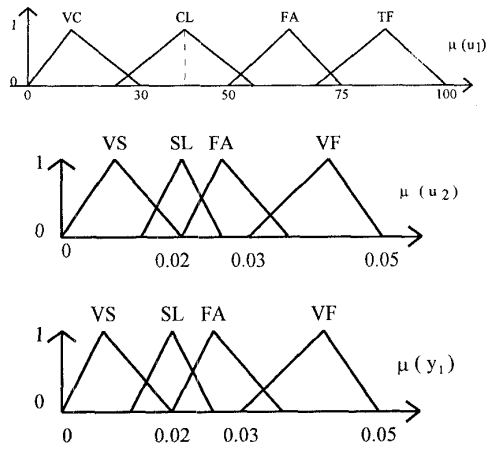


Figure 3 Fuzzy regions and the corresponding membership functions of u_1, u_2 and y_1

For its fuzzy rules, there are also two inputs u_1 which is the distance between the two robots, and u_2 which is the current speed of robot one, and output y_1 which is the speed of robot one for the next iteration. Figure 3 shows the fuzzy regions for these variables and their membership functions.

Intervals of definition u_1, u_2 and y_1 are each divided into four regions with linguistic variables **VC** (Very Close), **CL** (Close), **FA** (Far), **TF** (Too Far), and **VS** (Very Slow), **SL** (Slow), **FA** (Fast), **VF** (Very Fast) and **VS, SL, FA, VF** respectively. There are sixteen fuzzy rules. The fuzzy rule base can be formed as a 4 by 4 table with cells to hold the corresponding actions that must be taken given the conditions corresponding to u_1 and u_2 are satisfied. The table is initially empty. We can encode the input and output regions into bit-strings (of 0 and 1), for example, a coded bit-string of a fuzzy rule may look like :

VC	CL	FA	TF	VS	SL	FA	VF	VS	SL	F	VF
1	0	0	0	1	0	0	0	1	0	0	0

.

.

VC	CL	FA	TF	VS	SL	FA	VF	VS	SL	FA	VF
0	0	0	1	0	0	0	1	0	0	1	0

which means :

if distance = **VC** and current speed = **VS**
then next speed = **VS**

.

.

if distance = **TF** and current speed = **VF**
then next speed = **FA**

This string can be reduced considerably in length by noting that we need only encode the output control signal. The complete bit string has then the form:

VS	SL	FA	VF
1	0	0	0
VS	SL	FA	VF
0	1	0	0

.

.

VS	SL	FA	VF
0	0	1	0
VS	SL	FA	VF
0	0	1	0

made up of 16 individual output bit strings of length 4. This illustration is the encoding for the fuzzy knowledge base given in the next section.

The choice of output control signal to be set for each fuzzy rule is made by a genetic algorithm. It initialises randomly a population of complete bit-strings. Each of these bit-strings are then decoded into fuzzy rules and the associated fitness evaluated by the AFLC. The GA's cross-over procedure is modified to ensure that crossover takes place appropriately at a boundary of an individual output bit string (of length 4). The mutation operator is also modified. If the bit to be changed lies within an indiviudal output string and is not the bit currently turned on, that is, it has a value of 0, then it is turned on (set to 1), and all others in this output bit string are set to 0. If the bit is that one which is turned on, then the mutation process is continued by randomly selecting one of the other positions in this output string, turning it on and all the others off. Crossover probability was set at 0.6 and mutation rate at 0.01.

The GA then performs a self-directed search, learning better fuzzy rules for the second knowledge base of the AFLC. The fitness of each bit string, representing a set of fuzzy rules, is calculated by averaging the sum total of the distances between the robots for each iteration, during movement from their initial configurations to the target. If the robots collide then a penalty value is included in the fitness evaluation. This learning is seen to be performed quickly and automatically with no need for operational guidance other than the fitness values supplied to it by the AFLC.

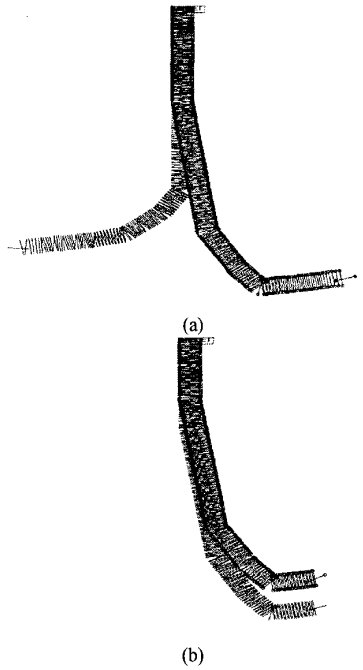
4. Simulation Results

We have trained the system from the following initial points for robot 2, (10,10), (30,10), (70, 10) with heading angles of -90°, 50°, 90°, 145°, 180°, 270°, and robot 1 from initial point (90, 10) with heading angle -90°, 50°, 90°, 145°, 270°. The following fuzzy knowledge base was obtained using fuzzy amalgamation, cf [1].

	VC	**CL**	**FA**	**TF**
VS	VS	SL	FA	TF
SL	VS	VS	FA	TF
FA	VS	SL	FA	FA
TF	VS	VS	FA	FA

With this limited training data the simulation results show that the proposed architecture is capable of generating fuzzy rules for the second layer of the AFLC system. The system is seemingly robust and it can guide the robots from surprisingly many initial positions in the work space to the target whilst avoiding each other. We present in Figure 4 two diagrams showing the trajectory of each robot from different initial states. In (a) robot one starts

from position (90,20) with heading angle 90^{o}, and robot 2 starts from position (10,20) with heading angle 90^{o}. Robot 2 reaches the target before robot 1 and it does not collide with robot 1 at any time. In (b) robot one starts from position (80,20) with heading angle -90^{o}, and robot 2 starts from position (80,10) with heading angle 90^{o}. The robots are initially close and are heading towards each other. Here robot 1 reaches the target first and does not collide with robot 2 at any time.



(a)

(b)

Figure 4 Trajectories of robots 1 and 2 using AFLC

Good results have been obtained from these initial configurations. This outcome is not true from all initial configurations, for consider those configurations set high in the upper right and left quadrants. Because of the limited number of grid points used in the learning process the GA will possibly have not learnt any appropriate rules in these areas. Increasing the number of test grid points used for learning the rules, will improve the ability of the architecture to learn the rules more effectively across the entire plane. This comes at a computational cost in increased GA learning and a significant increase in calculations in the amalgamation process. Such cost needs to be offset against the simplicity of the above construction and its computational cost to achieve reasonably good results. It is indeed surprising to observe that the fuzzy rules in the second layer work rather well from many configurations in the plane. This may be attributed to the fact that the initial grid selection consisted of points spread across the $y=10$ level, far from the target, and with a variety of heading angles.

A number of parameters influence the speed of convergence of the GA. These include the size of the populations, and the crossover and mutation probabilities. Research is currently being undertaken to examine how these parameters may be tuned to improve the performance of the integrated system.

5. Conclusion

An adaptive two layer Fuzzy Logic Control architecture is applied to the control of a multi robot system. A genetic algorithm was used to learn the second layer rule base and so expand an existing fuzzy knowledge control base in the first layer, making the system capable of adapting to a changing environment. The simulation results show that the architecture is seemingly robust. Further more :

- The system proposed here can be used in processes where a valid model is not available.
- The speed of learning using GAs is increased and the system can be modified if necessary later, by changing one of the knowledge bases or adding another layer to the system.

Acknowledgment

The first author is indebted to Edith Cowan University, Faculty of Science, Technology and Engineering for their research grant, Number A350-114.

References

[1] 'Integrating Fuzzy Logic and Genetic Algorithms for Intelligent Control and Obstacle Avoidance', by M. Mohammadian, R.J. Stonier, Second National Conference on Complex Systems, Rockhampton, Australia, pp 149-156, September 26-28, 1994.

[2] 'Intelligent Mobile Robot Path Planning with Fuzzy System Approaches', by J.X. Xu and K.C. Kin, IEEE International Workshop on Emerging Technologies and Factory Automation, Australia, Cairns , pp 28-41, September 1993.

[3] *'Collision-free trajectory control for multiple robots based on neural optimisation network'*, by J. Lee and B. Zeungnam, Robotica, Vol 8, pp 185-194, 1990.

[4] *'Real-time path-finding in Multi-robot Systems including Obstacle Avoidance'*, by E. Freund and H. Hoyer, International Journal of Robotics Research, Vol 7, 1, pp 42-70, 1988.

[5] *`Use of Liapunov Techniques for Collision-Avoidance of Robot Arms* , by R.J. Stonier, Advances in Control and Dynamic Systems, Vol. 35, Part 2, pp. 185-215, 1990.

[6] *`Collision-Avoidance of two PR Manipulators'*, by R.J. Stonier, Lecture Notes in Control and Information Sciences, Vol. 170, pp 248-264, 1992.

[7] *'Industrial applications of fuzzy control'*, Surgeon, (Ed), Elsevier Science Pub. Co., 1985.

[8] *'Neural networks and fuzzy systems'*, a dynamic system', by B. Kosko, Prentice-Hall, Englewood Cliff, 1992.

[9] *'Application of fuzzy set methodologies in industrial engineering'*, by G.W. Evans, W Karwowski, and M.R. Wilhelm, Elsevier, Netherlands, 1989.

[10] *'Fuzzy control systems'*, by A. Kandel, G. Langholz, CRC Press, USA, 1994.

[11] *'Development of fuzzy algorithms for servo systems'*, by Y.F Li, and C.C. Lan, IEEE Control System Magazine, Vol 9, No 3, pp 65-73, 1989.

[12] *'A linguistic self-organizing process controller'*, by W. Procyk, and E.H, Mamdani, Automatica, Vol 15, pp 15-30, 1979.

[13] *'Fuzzy logic control for steam generator feedwater control'*, by S. Rauj, J. Zhou, and R.A. Kisner, Proceedings of American Control Conference., San Diego, CA., May 1990, pp 1491-1493.

[14] *'Adaptive Hierarchical Fuzzy Controller'*, by G.V.S. Raju, J. Zhou, IEEE Transaction on Systems, Man, and Cybernetics, Vol 23, No 4, pp 973-980, July/August 1993.

[15] *'Generating Fuzzy Rules by Genetic Algorithms'*, by M. Mohammadian, X. Yu. R.J. Stonier, 3rd IEEE International Workshop on Robot and Human Communication : RO-MAN'94, Nagoya, Japan, July 18-20, 1994.

[16]' *Genetic Algorithms in search, optimisation and machine learning'*, by D. Goldberg, Addison Wesley, 1989