Soft Computing Paradigms for Learning Fuzzy Controllers with Applications to Robotics*

E. Tunstel†, M.-R. Akbarzadeh-T, K. Kumbla and M. Jamshidi
NASA Center for Autonomous Control Engineering
Department of Electrical and Computer Engineering
University of New Mexico
Albuquerque, NM 87131 USA

Abstract

Three soft computing paradigms for automated learning in robotic systems are briefly described. The first employs Genetic Programming (GP) to evolve rules for fuzzy-behaviors to be used in mobile robot control. The second paradigm develops a two-level hierarchical fuzzy control structure for flexible manipulators. It incorporates Genetic Algorithms (GA) in a learning scheme to adapt to various environmental conditions. The third paradigm concentrates on a methodology that uses a Neural Network (NN) to adapt a fuzzy logic controller (FLC) in manipulator control tasks. Simulation results of fuzzy controllers learned with the aid of these soft computing paradigms are presented.

1 Introduction

Traditional methods which address robotics control issues rely upon strong mathematical modeling and analysis. The various approaches proposed to date are suitable for control of industrial robots and automatic guided vehicles which operate in *structured* environments and perform simple repetitive tasks that require only end-effector positioning or motion along fixed paths. However, operations in unstructured environments require robots to perform more complex tasks for which analytical models for control can often not be determined. In cases where models are available, it is questionable whether or not uncertainty and imprecision are sufficiently accounted for. Under such conditions fuzzy logic control is an attractive alternative that can be successfully implemented on real-time complex systems. Fuzzy controllers are robust in the presence of perturbations, easy to design and implement, and efficient for systems that deal with continuous variables [1]. In many practical instances, fuzzy control alone is not sufficient for addressing the complex intelligent control problems of robotic systems. Additional tools are necessary to achieve adaptation and learning capabilities. The control schemes described herein are examples of approaches that augment fuzzy logic with soft computing paradigms to achieve the level of intelligence required of complex robotic systems.

Three fuzzy control approaches are described. The first incorporates fuzzy logic control, with rule learning by genetic programming (GP) [2], into the framework of behavior control for mobile robot navigation. The second develops a two-level hierarchical fuzzy control structure for flexible manipulators. It incorporates genetic algorithms (GA) [3] in a learning scheme to adapt to various environmental conditions. The third concentrates on a methodology that uses a Neural Network (NN) to learn rules or change membership functions for a fuzzy logic controller (FLC) in a two rigid link control.

2 Rule Learning for Mobile Robots

In the fuzzy-behavior control scheme, a mobile robot behavior is encoded as a fuzzy rule-base designed to map relevant sensor inputs to control outputs according to the desired control policy. In developing fuzzy-behaviors we take into consideration the notion that humans may not have the best solutions for designing knowledge-based controllers with interacting rule-bases. As an alternative, good results have been achieved by employing genetic programming to learn a subset of the fuzzy-behaviors [4] that comprise a given behavior control system. In GP, a population is comprised of computer programs (individuals) that are candidate solutions to a particular problem. These individuals participate in a simulated evolution process wherein the population evolves over time in response to selective pressure induced by the relative fitnesses of the individuals in a particular problem environment. In our approach, each program executes condition-action statements which collectively serve as a rule-base to be embedded in a fuzzy-behavior. In the process of learning fuzzy control rules, GP ma-

*This work was supported in part by NASA under contract # NCCW-0087.

†On leave from Jet Propulsion Laboratory, Pasadena, CA.

nipulates the linguistic variables directly associated with the behaviors and enhances diversity of a population of behaviors by allowing for rule-sets of various sizes.

Given a desired motion behavior, the behavior search space is contained in the set of all possible rule-bases that can be composed recursively from a set of *functions* and a set of *terminals*. The function set consists of components of the generic *if-then* rule and common fuzzy logic connectives, i.e. functions for antecedents (ANT), consequents (CONSQ), fuzzy intersection (f_AND), rule inference (IF-THEN), and fuzzy union (f_OR). The function f_OR takes a variable number of arguments (equal to the number of rules) and the remaining functions each take two arguments. The terminal set is made up of the input and output linguistic variables and the corresponding membership functions associated with the problem.

2.1 Example

We applied GP for developing a mobiile path-tracking behavior. Its task was to learn/evolve fuzzy rules to properly steer a mobile robot for path following in the plane. The problem is taken from Hemami [5] where it is formulated for a class of low-speed (less than 2 m/s) tricycle-model vehicles. The state vector consisted of measurable pose errors associated with path following. The position error (ε_d) is taken as the deviation from the nearest point on the desired path. The orientation error (ε_θ) is the angular deviation of the robot's heading from the tangent to the desired path. The front wheel steering angle (δ) is the corrective control action that causes the error states to decay to zero, thus forcing the robot to follow the path. Thus, the rule-base to be learned is a two-input-one-output fuzzy-behavior that will map the error states into a steering angle at each time step. In our GP applications we focus the effort on evolving the rule-base and assume that the membership functions are specified *a priori* and are fixed. Given the function and terminal sets, a rule-base that could potentially evolve via GP can be expressed as a parse tree with preordered branches. An example of a syntactically valid rule-base of two rules is depicted in Figure 1 along with its interpretation as a linguistic rule-base. The linguistic notation in the figure is interpreted as follows: $NB \equiv$ "negative big", $NS \equiv$ "negative small", $Z \equiv$ "zero", $PS \equiv$ "positive small", $PB \equiv$ "positive big", and lowercase prefixes "p" and "o" designate fuzzy membership functions for position error and orientation error respectively. Fuzzy sets for the steering angle are labeled without a prefix.

During the GP evolution process, each rule-base in the current population is evaluated via simulation to determine its fitness for tracking the desired path. Eight fitness cases (initial conditions) were used for this example. The fitness of a given rule-base was based on the Euclidian norms of the error state vector at the end of each fitness case. Results from a simulation of a path-

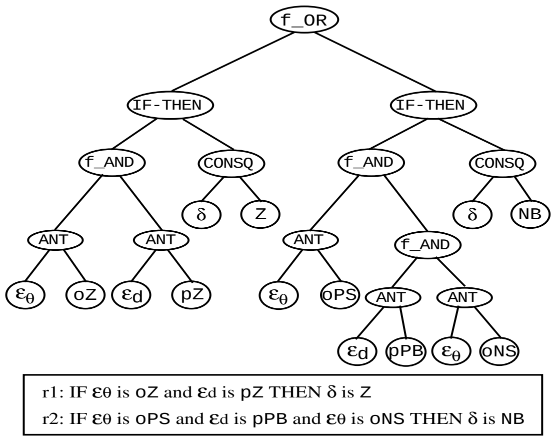

Figure 1: **Rule-base tree structure.**

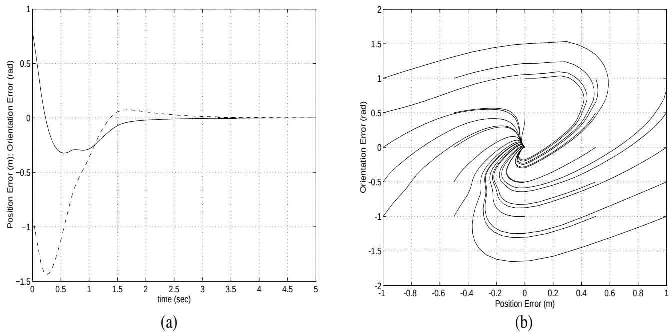

Figure 2: **Path-tracking performance: (a) ε_d—, ε_θ- -; (b) phase portrait.**

tracking behavior learned by GP is shown in Figure 2a. The figure illustrates the fuzzy-behavior performance in terms of pose errors with initial values of ε_d = 0.8 m. and ε_θ = -0.9 rad. Improved results can be obtained by modifying the fitness function used to drive the evolution process. This was demonstrated in [4] where additional results and more detailed discussion can be found. Although the evolved behavior learned the control rules using only eight pre-selected fitness cases, it was able to generalize when started from initial conditions throughout the error state space. This is shown in the phase portrait of Figure 2b which reveals that the origin is a stable node of the system.

3 GA-learning Fuzzy Control for Flexible Robots

Flexible robots are classified as distributed parameter systems (DPS) and are functions of space as well as time. Due to the complexity of a mathematical representation for such systems, fuzzy logic is considered an attractive alternative to their control. One of the issues in development of fuzzy controllers is determining a faithful expert knowledge. An expert knowledge, however, is dif-

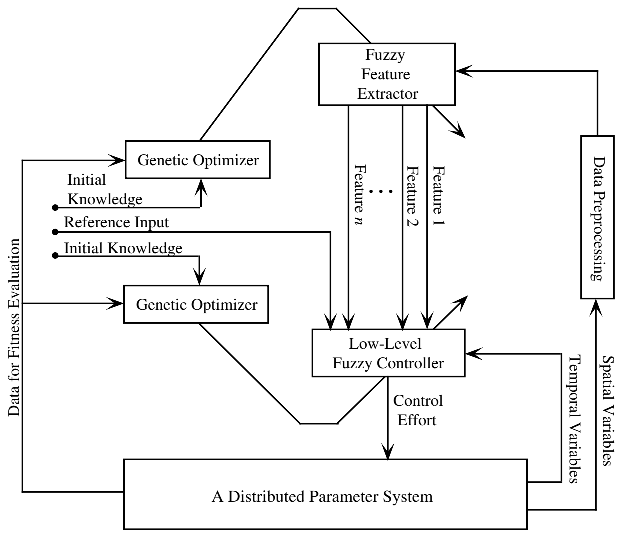

Figure 3: **GA-Based Learning Hierarchical Control Architecture**

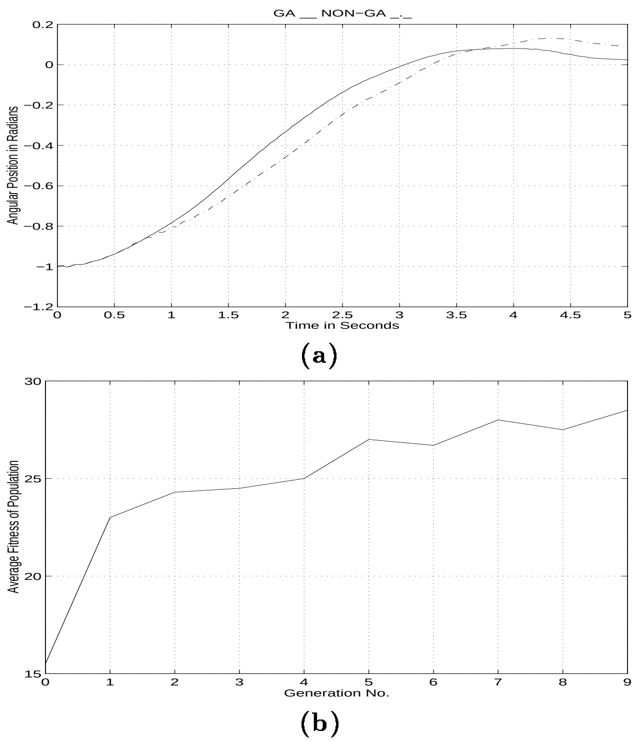

Figure 4: **GA Simualtion (a) Comparison of Time Responses (b) Plot of Average Fitness**

ficult to produce since there is often no human expert to be consulted and training a human expert may not be a feasible alternative due to cost and other practical considerations. Furthermore, human psychological issues may prohibit a faithful reproduction of rule-base from an expert. In addition, the unstructured operating environments such as in space and waste handling projects require the robot controller to also adapt to changing conditions. As an alternative, good results have been achieved by employing genetic algorithms to tune parameters within a fuzzy controller's knowledge base [6]. Genetic algorithms equip the Fuzzy controller with some evolutionary means by which it can improve its rule-base when faced with inadequate a-priori expert knowledge or varying circumstances in its operating environment.

Figure 3 shows a possible GA-learning hierarchical fuzzy control architecture. Within the hierarchical control architecture, the higher level module serves as a fuzzy classifier by determining spatial features of the arm such as *straight*, *Oscillatory*, *Curved*. This information is supplied to the lower level of hierarchy where it is processed among other sensory information such as errors in position and velocity for the purpose of determining a desirable control input (torque). In [7] this control system is simulated using only a-priori expert knowledge. In the given structure, a genetic algorithm fine tunes either parameters of membership functions and/or rule set's antecendents and consequents. If the parameter space includes both parameters from membership functions as well as rules, it can involve more degrees of freedom than is necessary to uniquely represent a given nonlinear mapping between inputs and outputs of the controller. Consequenly, in this paper, the GA genetic representation involves only membership parameters.

3.1 Example

To demonstrate the potential usage of genetic algorithms, GA is applied to optimize parameters related to input membership functions of the higher level of hierarchy as is shown in Figure 3. Other parameters in the knowledge base are not allowed to vary. The following fitness function was used to evaluate various individuals within a population of potential solutions,

$$fitness = \int_{t_i}^{t_f} \frac{1}{e^2 + x^2 + 1} dt, \qquad (1)$$

where e represents the error in angular position and x represents overshoot. Consequently, a fitter individual is an indivual with a lower overshoot and a lower overall error (shorter rise time) in its time response.

When developing the hierarchical controller, some initial knowledge is expected to be supplied through expert knowledge for feature extraction and the lower level control. The initial population is made up of parameters chosen randomly with the same mean as the a-priori parameters. Figure 4.a shows the time response of the GA-optimized controller when compared to previously obtained results through the non-GA fuzzy controller. The rise time is improved by 0.34 seconds (an 11% improvement), and the overshoot is reduced by .07 radians (a 54% improvement). Figure 4.b shows the average fitness of each generation. A total of 10 generations were

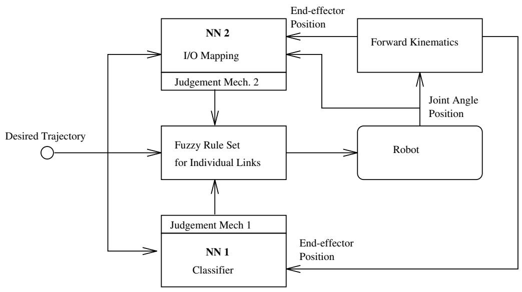

Figure 5: Schematic of Neuro-Fuzzy Controller

simulated. Mutation rate for this simulation was set at 0.1. Probability of cross over was set to 0.6.

4 Self-Organizing Neuro-Fuzzy Controller

The learning capability of the neural network can be made use of in designing the fuzzy controller. The *Self-organizing fuzzy controller* is one such combination of a neural network and a fuzzy controller [8]. Figure 5 shows a schematic diagram of the system forming the self-organizing fuzzy controller. The aim of this system is to automatically form the fuzzy controller. It uses two neural networks of the *back propagation learning type*, NN1 and NN2. NN1 acts as a classifier of the dynamic responses of the system being controlled (robot system). NN2, set in judgment mechanism 2, has knowledge of the dynamic characteristics of the object system. Judgment mechanism 2 has a self tuning mechanism to automatically determine the normalizing values of the membership functions to control the object system adequately.

In this particular case NN1 classifies the error in the joint angle positions of a robot system to several typical patterns such as a similar pattern to the desired response or an oscillating and diverging pattern or an oscillating and slowly converging pattern or any other pattern. The result of the classification is sent to judgment mechanism 1. NN2 is made to learn the dynamic characteristics of the object system through pairs of input and system response. NN2 can then be used to simulate the object system in cases where it is too risky to control the object system with an incomplete fuzzy controller.

4.1 Example

Auto tuning neuro-fuzzy controller is applied to control a model of two link robotic manipulator [9]. In this case, since the model of the object system is known, NN2 is not utilized.

A step change of one radian is applied as the desired input to the two joints of a robotic manipulator. The temporal values of the joint angle is feed to a neural network (multi-layered perceptron). This neural network

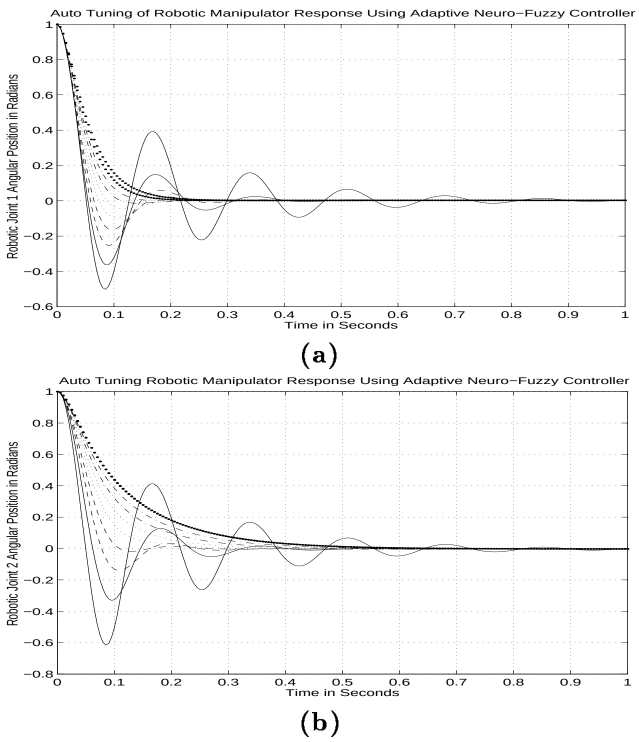

Figure 6: **Auto Tuning of Robotic Manipulator Response: (a) Joint One (b) Joint Two**

consisting of 20 input nodes, 10 nodes in the hidden layer and 3 output nodes is initially trained to recognize three different patterns of responses. They are oscillatory but converging, exponentially converging and diverging responses. The output of the NN ranges from 0 through 1, depending on the degree of these responses, for example if the oscillation persists for a long time then the output may be closer to 1.

Figure 6 shows the response of the system. Initially the fuzzy PD controller is not tuned. The solid lines indicates an oscillatory but converging response. The NN output as shown in Figure 7 shows the degree of the oscillatory response shown with solid lines. The adaptation mechanism acts by changing the scaling factor of the derivative portion of the fuzzy PD controller. In the subsequent responses shown by the dashed lines shows less oscillations. Finally after 10 iterations the oscillations die down (dotted lines). The controller is tuned to perform in a desired manner. The desired response here is exponentially coverging. Figure 7 shows the degree oscillations is reducing as the degree of exponentially converging response increases (dashed line).

5 Conclusions

This paper shows applications of three softcomputing paradigms for knowledge enhancement of fuzzy logic controllers. In the context of mobile robot control, a fuzzy

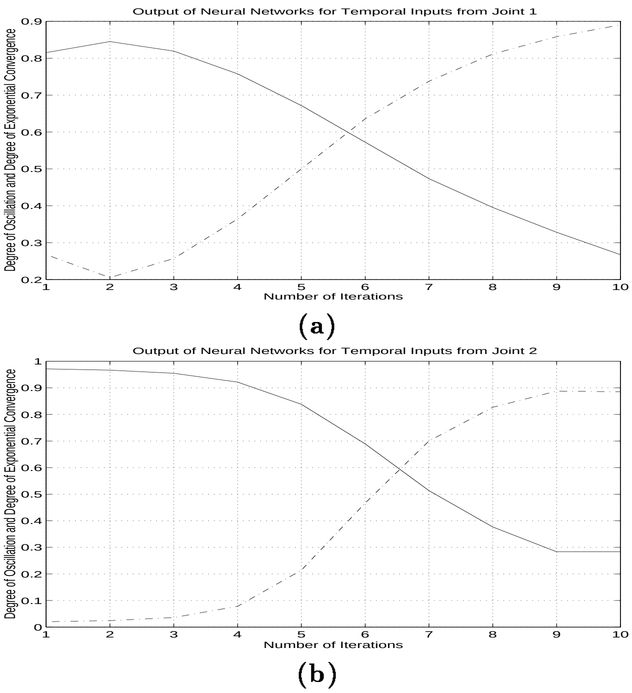

Figure 7: **Degree of Oscillations and Exponential Convergence: (a) Joint One (b) Joint Two**

logic-based system has the advantage that it allows the intuitive nature of sensor-based navigation to be easily modeled using linguistic terminology. Moreover, precise sensing and detailed maps of the environment are not absolutely necessary. The hierarchy of distributed fuzzy behaviors provides an efficient approach for controlling mobile robots, or other complex systems, that have many rules. The practical utility of the control scheme lies in its decomposition of the totality of rules into subsets of rules that are consulted only when applicable. When the state of the world satisfies the conditions for activation of a single behavior, or several, there is no need to process rules from behaviors that do not apply (as in the conventional FLC architecture). Processing rules from irrelevant behaviors would result in unnecessary consumption of computational resources and possible introduction of "noise" into the decision-making process. The hierarchical decomposition of behavior leads to a suitable framework for situated adaptation and is amenable to design by evolutionary methods.

Second issue in this paper is the GA-learning fuzzy controller. The controller offers a hybrid fuzzy approach to control of flexible arms. The second level hierarchy determines the fuzzy features of *Straight, Oscillatory, and Gently Curved.* At this point, only input membership functions for the higher level hierarchy were allowed to adapt. Even with this limited adaptation, response has shown significant improvement in both decreased rise time and reduced overshoot. Greater improvements are expected if all membership functions are allowed to adapt. However, too many adaptation parameters can cause extra degrees of redundany which complicate the search space and increase computational requirements. Hence, rule adaptation will be considered separately from membership function adaptation in the next stage of this research.

In the third focus of the paper an adaptive self-organizing fuzzy controller is proposed which can automatically tune the fuzzy rule base and also create new fuzzy rules by the help of neural networks. The proposed scheme has now a learning mechanism by which it can adapt to changing conditions. The neuro-fuzzy controller is applied to a two-link robotic manipulator. The simulation results show the fuzzy controller is properly tuned to perform towards the desired specifications.

References

[1] E. H. Mamdani, "Twenty Years of Fuzzy Control: Experiences Gained and Lessons Learnt", *IEEE Intl. Conf. on Fuzzy Systems*, pp. 339-344, 1993.

[2] J. R. Koza, *Genetic Programming: On the programming of Computers by means of natural selection*, MIT Press, Cambridge, MA, 1992.

[3] D. E. Goldberg,"Genetic Algorithms in Search, Optimization and Machine Learning," *Addison-Wesley*, MA, NY, 1989.

[4] E. Tunstel and M. Jamshidi, "On Genetic Programming of Fuzzy Rule-based Systems for Intelligent Control", *Intl. Jrnl. of Intelligent Automation & Soft Computing*, Vol. 2 No. 2, 1996, in press.

[5] A. Hemami "Steering Control Problem Formulation of Low Speed Tricycle-model Vehicles", *International Journal of Control*, Vol. 61, No. 4, pp. 783-790, 1995.

[6] M. A. Lee and H. Takagi, "Integrating Design Stages of Fuzzy Systems Using Genetic Algorithms," *Proceedings of the IEEE International Conference on Fuzzy Systems*, San Francisco, CA, pp. 612-617.

[7] M.-R. Akbazadeh-T., M. Jamshidi, and N. Vadiee, "A Hierarchical Fuzzy Controller Using Line-Curvature Feature Extraction For A Single Flexible Arm,"*Proc. of the 1994 IEEE Conference on Fuzzy Systems, FUZZ'94*, pp 524-529, Orlando, Fl, 1994.

[8] K. Kumbla M. Akabarzadeh and M. Jamshidi, "*TMS320 DSP Chip Based Neuro-Fuzzy Controller*" IEEE Conference on Man, System and Cybernetics, Vancouver, pp 4015-4020, October 1995.

[9] K. Kumbla and M. Jamshidi, "Control of Robotic Manipulator Using Fuzzy Logic", *IEEE Intl. Conf. on Fuzzy Systems*, pp 518-523, Orlando, FL, 1994.