World Academy of Science, Engineering and Technology
International Journal of Mechanical and Materials Engineering
Vol:6, No:6, 2012

Evolutionary Algorithms for Learning Primitive Fuzzy Behaviors and Behavior Coordination in Multi-Objective Optimization Problems

Li Shoutao, Gordon Lee

Abstract**—**Evolutionary robotics is concerned with the design of intelligent systems with life-like properties by means of simulated evolution. Approaches in evolutionary robotics can be categorized according to the control structures that represent the behavior and the parameters of the controller that undergo adaptation. The basic idea is to automatically synthesize behaviors that enable the robot to perform useful tasks in complex environments. The evolutionary algorithm searches through the space of parameterized controllers that map sensory perceptions to control actions, thus realizing a specific robotic behavior. Further, the evolutionary algorithm maintains and improves a population of candidate behaviors by means of selection, recombination and mutation. A fitness function evaluates the performance of the resulting behavior according to the robot's task or mission. In this paper, the focus is in the use of genetic algorithms to solve a multi-objective optimization problem representing robot behaviors; in particular, the A-Compander Law is employed in selecting the weight of each objective during the optimization process. Results using an adaptive fitness function show that this approach can efficiently react to complex tasks under variable environments.

Keywords**—**adaptive fuzzy neural inference, evolutionary tuning

I. Introduction

WHILE several approaches have been suggested for designing nonlinear controllers, the problem becomes more complex when plant uncertainties and noise are considered. An approach that has gained some success relies on a non-parametric philosophy whereby a fuzzy block is used to handle uncertainties and imperfections while a neural network block addresses the underlying model dynamics.

The classical adaptive neural-network based fuzzy inference system (ANFIS) approach [1] is such architecture and generally provides good overall system performance when the control gains are properly selected; however, this may not always be feasible, particularly when the variability or uncertainties are unknown.

One can employ a set of ANFIS blocks to form a generalized ANFIS (GANFIS) that can approximate a nonlinear structure [2] (Figure 1). In the GANFIS design, the idea is to represent the desired control action by a transfer function approximation as:

Li Shoutao is with the College of Communication Engineering; Jilin University; Changchun, 130025, China. email: list@jlu.edu.cn

Gordon Lee is with the Dept. of Electrical & Computer Engineering; San Diego State University; 5500 Campanile Drive; San Diego, California, 92182-1309. email: glee@mail.sdsu.edu

$$H(s)=\frac{\sum_{\ell=0}^{m}\beta_\ell s^\ell}{\sum_{\ell=0}^{n}\alpha_\ell s^\ell}=\sum_{j=1}^{m-n}\gamma_j s^j+\gamma_0+\sum_{j=1}^{n}\frac{\delta_j}{s+a_j} \quad (1)$$

where m is the order of the numerator and n is the order of the denominator of the transfer function approximation to a nonlinear function of the output error, $f(e)$, y_d is the desired output and y is the actual output.

The control law is:

$$u(s)=\sum_{i=1}^{n_r}\overline{\omega}_i\Big[\sum_{j=1}^{m-n}p_{i,j}s^j E(s)+p_{i,0}E(s)+\sum_{j\neq 1}^{m}\frac{p_{i,j}}{(s+a_j)}E(s)\Big] \quad (2)$$

where $p_{i,j}$ are the consequent parameters of the ANFIS blocks. In designing the GANFIS controller as well as other versions of an ANFIS-based architecture, the issue is proper selection of the consequent and premise parameters.

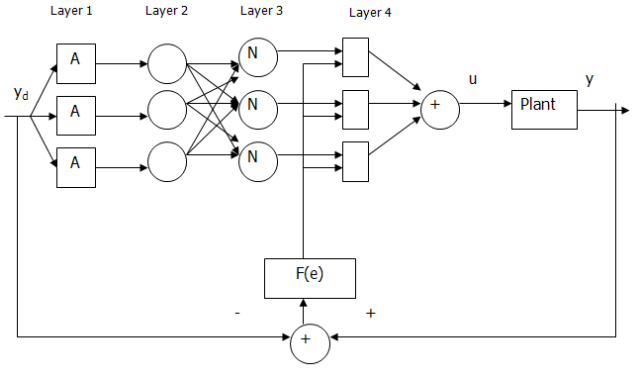

Fig. 1 The Generalized ANFIS Controller

It is shown in [3] that evolutionary algorithms may be used in selecting the GANFIS parameters on-line, resulting in a more stable structure. Thus, one must select the evolutionary parameters (the mutation and crossover probabilities) and the fitness function, all of which have an impact on the evolutionary process.

In this paper, we focus on the fitness function and in particular, how multi-objective functions can be addressed in the GANFIS architecture, which is the main contribution of this paper. Section II provides a brief development of the fitness function using heuristics and prediction as well as the

incorporation of simulated annealing. Then, in Section III, we present the multi-objective function format and show how this can be incorporated into the GANFIS. In particular, the A-Compander Law may be used to tune the weights of each objective function. Section IV presents a nonlinear system example, subjected to noise and parameter variation. Results show the feasibility of employing the GANFIS controller to nonlinear systems with noise or system variation.

II. THE GENERALIZED ANFIS CONTROLLER STRUCTURE

For completeness sake, a brief summary of the generalized ANFIS (GANFIS) controller is provided. The GANFIS, shown in Figure 1, requires the selection of both premise and consequent parameters; here we use evolutionary methods in selecting these parameters as shown in Figure 2. In most evolutionary approaches, genetic searching is used which consists of a finite repetition of three steps at each generation: selection of the parent chromosomes for the next generation (usually an elitist selection for a percentage of the next generation), recombination using crossover and mutation operations [4], and a fitness function that describes the *goodness* of individual members of each generation. In [5], Rajapakse and others employ evolutionary algorithms to tune fuzzy logic controllers, but then use an on-line neural network model of the process as a separate block. We use the evolutionary learning as *part* of the adaptive neural network fuzzy inference controller, rather than separate each operation (evolutionary tuning, fuzzy logic controller, neural network model of the plant) in the design process. Further, the parameters of the evolutionary learning operation (population size, mutation operator, cross-over operator and fitness function) are adaptively changed based upon the overall system performance measure. Pedrycz [6] states that the mutation rate and the crossover rate can be experimentally adjusted from results from a series of observations of past simulation and provides a method using Fuzzy meta-rules.

The evolutionary module runs several generations of candidate premise and consequence parameter chromosomes and selects the best set, according to a fitness function of the form:

$$F = \sum_{i=1}^{n_r} e_i^2 \tag{3}$$

where the error is the difference between the desired output and the actual output.

In order to improve the fitness value, a fitness function based upon current, past and expected future values can be used.

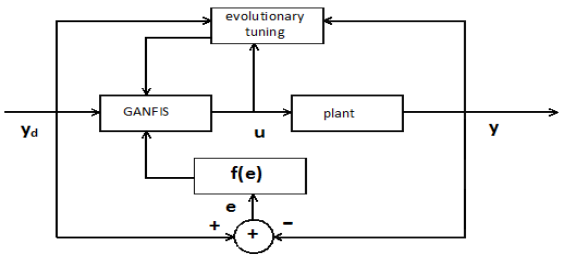

Fig. 2 GANFIS with Evolutionary Tuner

For example, in [7] a heuristic fitness function is developed that is based upon past, current and *future* knowledge information using the form:

$$f(n)= g(n) + h(n) + q(n) \tag{4}$$

where g(n) is the classical fitness function based upon an error metric using current information, h(n) is the fitness function component based upon some historical information related to a *predicted* target and q(n) is a heuristic function based upon *expected future* knowledge. We differentiate between prediction, based upon deterministic historical data and future heuristics, based upon probabilistic information.

To improve performance, in [8] we integrated simulated annealing (SA) in the evolutionary process (EP). Simulated annealing is an optimization process in which a function of a parameter (temperature) is used to decide upon an uphill move. Here we integrate SA in the EP in the following way: during the generation of a new population, typically half of the population is kept (elitism) while the bottom half is replaced by a random set of new chromosomes. Here, instead of using a simple random generation, we mutate the bottom half of the population (the neighborhood function in SA) and then perform SA in generating the new half of the population. Tuning is performed in selected the SA parameters (temperature decay factor) based upon performance at each generation. In [8], we show that this modification improves system performance, even under noise and parameter variation.

The issue, then, is how to add several objectives to the fitness function. In [9], fuzzy weights are used for a multiplicative fitness function, made up of different objectives, and the focus was on performance of the fitness function. Here we focus on additive objectives and wish to investigate the notion of using fuzzification in selecting the weights and then on how it is used in the controller. That is, in this paper, we focus on how the fitness function made up of many objectives can be incorporated into the GANFIS structure.

III. TUNING THE EVOLUTIONARY PARAMETERS

The mutation and crossover rates are two important evolutionary parameters and are typically statically set through trial-and-error in classical evolutionary algorithms [10].

In [11], the effects of the crossover rate P_c and mutation rate P_m to the maximum fitness and average fitness values are discussed. The larger the error is between the fitness values of two individuals, the stronger is the degree of the mutation rate and crossover rate. In [12], the mutation and crossover rate are tuned using different functions of the current fitness values. For example, one may select:

$$p_C = \frac{|f_i(n) - \bar{f}(n)|}{\bar{f}(n)} \tag{5}$$

$$p_m = \frac{[\bar{f}(n) - f_i(n)]}{2 * \bar{f}(n)} \quad (6)$$

where $f_i(n)$ is the current fitness function associated with the chromosome that requires the crossover or mutation operation and $\bar{f}(n)$ is the maximum fitness function for generation *n*. Notice that (5)-(6) do not require any *a priori* knowledge in selecting the probabilistic rates; rather the estimators simply use current fitness values at each generation.

In [13], we use simulated annealing with an adaptive tuning factor to improve convergence. In this paper, the focus is on how the fitness function can be altered to incorporate multi-objectives based upon robot behavior using fuzzification of the weights. Each objective function may be associated with a different robot goal such as tracking a desired path or using the minimum energy or traversing a maze in minimum time. How these objective functions are incorporated into a single fitness function is now discussed.

IV. THE MULTI-OBJECTIVE FITNESS FUNCTION USING FUZZY BEHAVIOR

A drawback of the classical approach to multi-objective function optimization is the issue of how to select the weights of each objective. The A-Law Compander function [14] can alleviate these problems by using a compact formulation of fuzzification.

The basic A-Law Compander equation in the compressor mode is given as:

For $0 \le (|x| / xmax) \le 1/A$:

$c(x) = 1 + sgn(x) * A |x| / \ln(A)$

For $1/A \le (|x|/xmax) \le 1$:

$c(x) = xmax * sgn(x) * (1 + \log(A |x|/xmax)) / (1 + \ln(A))$ (7a)

The A-Law Compander equation in the expander mode is given as:

For $x \le (1 / \ln(A))$:

$c(x) = x (1 + \ln(A)) / A$

For $x > (1/\ln(A))$:

$c(x) = \exp(x (1 + \ln(A)) - 1) / A$ (7b)

where xmax is the largest value of the input x, c(x) is the fuzzified output, and ln (.) is the natural log function. It is assumed that the value of A is greater than unity and each A-Compander function describes a fuzzification operation.

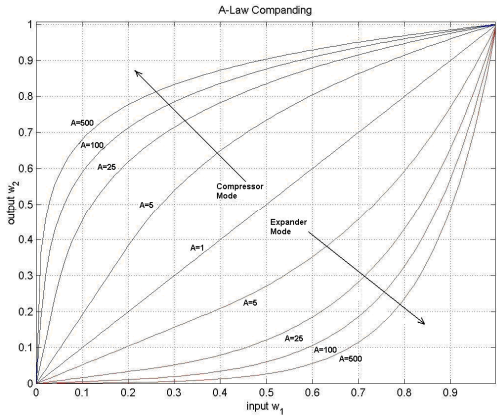

Fig. 3 A-Law Compander Functions

The approach here is to employ the A-Law Compander function to fuzzify the weights of each objective in a multi-objective fitness function. Each weight is fuzzified through an A-Law Compander function that maps the weight with bounds [min, max] to the range [0, 1] according to the function definition. Hence the number of parameters that need to be selected for fuzzification reduces the number of premise parameters by a factor of two (the average and standard deviation for a Gaussian membership function, for example) to just the number of membership functions desired (using the A-Compander form of the membership function)

As the value of A approaches unity from the expander side of Figure 3, the priority of the rule, which is mapped to its membership function, increases. The priorities of rules are adaptively changed using different values of A, based upon the current epoch and also their performance in the current epoch as compared to their performance in the previous epochs. The weights given to each objective are in the range [0,1].

Using this convention, the expander side of the A-Law Compander is chosen for the selection of the weight for each objective in the multi-objective fitness function. A method which can be used to update the value of A assigned to each rule at the end of each epoch needs to be found. In this paper, the strategy is to update the value of A at the end of each epoch, when the present fitness value is compared with the corresponding fitness value in the previous epoch. The fitness is typically selected as the square-root of the sum of the squared errors between the desired output values and the current output values. A modification of the fitness function is performed by incorporating multiple objectives with fuzzy weights using this fuzzification approach is employed here, which is the main contribution of the paper.

If the current fitness value f(n) is lower than or equal to the fitness value in the previous epoch, f(n-1), then the value of A(n+1) for each rule is modified for the next epochs using the equation:

$$A(n+1) = A(n) + d(n) * k \quad (8)$$

where A is the compander curve value assigned to a particular rule, and k is a scaling factor. In our case, the value of A corresponds to the weight associated with each objective. An increasing or decreasing weight is selected based upon the size of d and the epoch number; other scaling factors may be used. In this paper, the value d is selected as the difference between the minimum and maximum value of the fitness function, normalized to the maximum value. Notice that as the error between the fitness function values decrease, the scaling factor increases. But the net result goes to zero as the value of the A-Compander function converges [14]. To improve tuning, the classical fitness function is modified by this fuzzification approach for each objective and is illustrated in the example below.

That is, consider a fitness function:

$$f = \sum_{i=1}^{q} a_i f_i \tag{9}$$

where each f_i represents a particular objective and q is the number of objectives in the multi-objective fitness function. The coefficient a_i represents the weight on each objective. The problem then is to choose the coefficients $\{a_i\}$ such that the objectives are met while minimizes the effects of parameter variation and noise. In this paper, a fuzzy tuning approach is employed using the A-Compander Law of (8).

V. Example

Consider the nonlinear system [15]:

$$\begin{aligned} \dot{x}_1(t) &= x_2^3(t) + u(t) \\ \dot{x}_2(t) &= u(t) \\ y(t) &= x_1(t) \end{aligned} \tag{10}$$

where y(t) is the output and u(t) is the control input. It is desired that the output track the function:

$$y_d(t) = \sin 2t * e^{-1.5t} \tag{11}$$

The control parameters for the tests were selected as follows: the population size of 20, six membership functions in the ANFIS block, four bits for each chromosome with an elitist selection. Consider the cases when the coefficient associated with the nonlinearity go from unity to another value (parameter variation); four cases have been studied as summarized in Table I.

The coefficient represents the new value of the coefficient associated with the nonlinear state in the dynamics (10), i.e., from unity while the third column defines when the variation occurred. The input noise is at 20 db. The multi-objective function studied here is:

$$f_1 = \sum[y_d(n) - y(n)]^2$$

$$f_2 = \sum[u(n)]^2 \tag{12}$$

and then the multi-objective fitness function is a linear combination of f_1 and f_2 with appropriate weights as in (9). Here n is the index of the sampled time trajectory.

TABLE I
Case Studies in Parameter Variation

Case #	*Coefficient*	*Time of Change*
1	0.01	2 sec
2	100	2 sec
3	0.01	0.5 sec
4	100	0.5 sec

First, we look at the case when no fuzzified weights are used. That is, each objective function is simply added together. The results for the four test cases are as shown, including the error trajectories. Figures 4-7 provide typical results. Note that even without fuzzy tuning of the weights of each objective, there is reasonable performance under parameter variation and noise.

Then consider using the approach developed here where the weights for each objective functions are selected using the A-Compander Law for fuzzification. Figures 8-11 show the results.

It is interesting to note that with the addition of fuzzification of the weights for each objective function through (7b) as the adaptive tuning, the error trajectories are either smaller or have better settling times. Thus, by comparing performance during the evolution, this parameter can be appropriately adjusted to improve performance.

VI. Conclusions

It is shown that performance can be improved by appropriately selecting the weights of each objective in a multi-objective optimization problem. In this paper, tuning by fuzzification using the A-Compander Law is suggested. In the future, we plan to implement the approach on real systems such as robotic colonies as well as investigate convergence properties in a formal way.

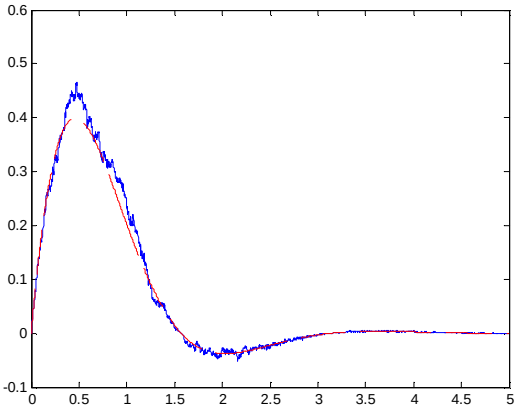

Fig. 4 Desired Output versus Actual Output for Case 1: No Fuzzy Tuning

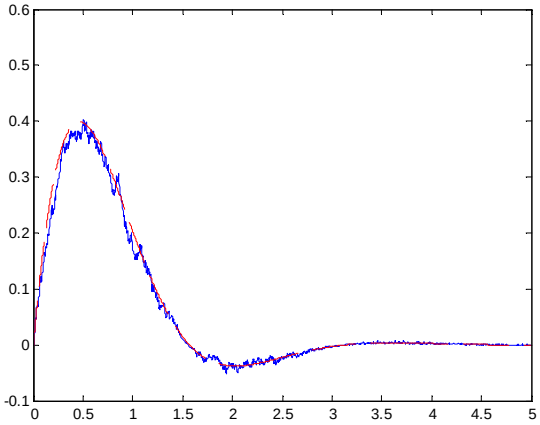

Fig. 5 Desired Output Versus Actual Output for Case 2: No Fuzzy Tuning

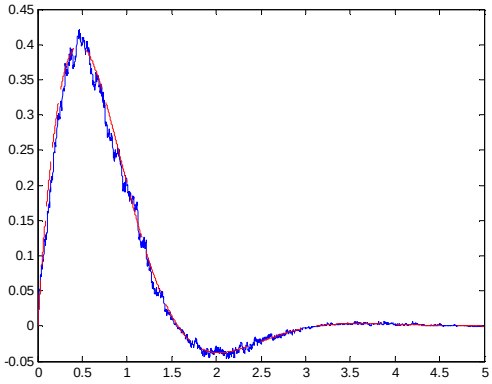

Fig. 6 Desired Output Versus Actual Output for Case 3: No Fuzzy Tuning

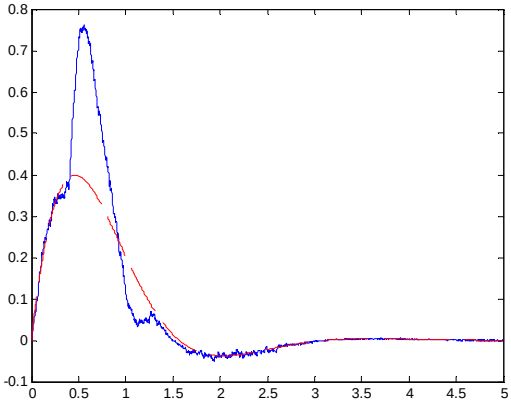

Fig. 7 Desired Output versus Actual Output for Case 4: No Fuzzy Tuning

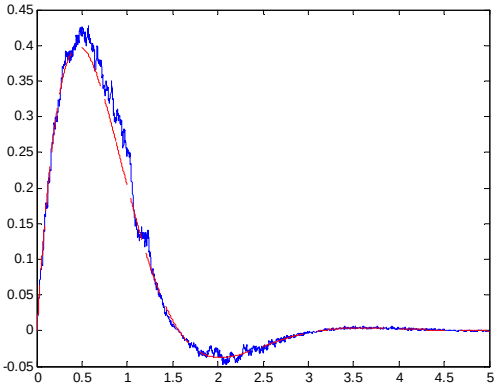

Fig. 8 Desired Output versus Actual Output for Case 1: with A-Law Yuning

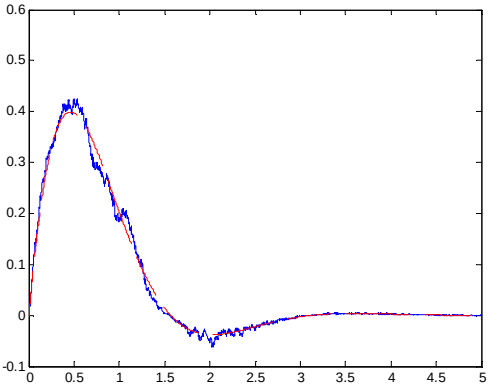

Fig. 9 Desired Output versus Actual Output for Case 2 with A-Law Tuning

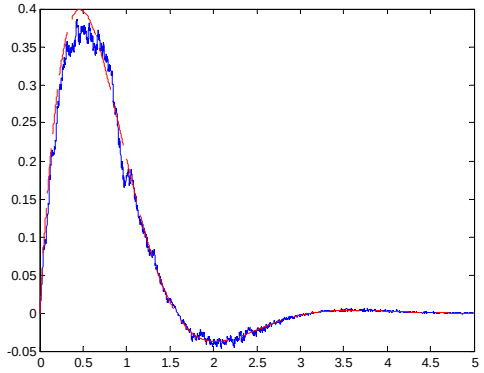

Fig. 10 Desired Output Versus Actual Output for Case 3 with A-Law Tuning

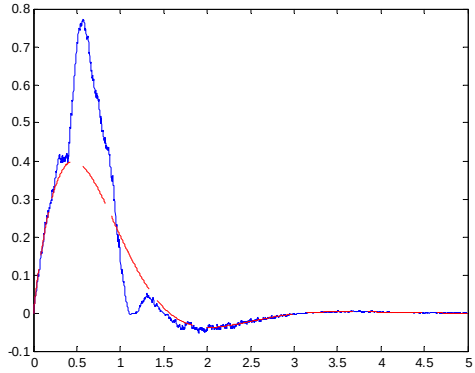

Fig. 11 Desired Output versus Actual Output for Case 4 with A-Law Tuning

REFERENCES

[1] J.S.R. Jang, "ANFIS: Adaptive-Network Based Fuzzy Inference System", IEEE Transactions on Systems, Man and Cybernetics, Vol. 23, No. 3, pp. 665-685, 1993.

[2] G. Lee and E. Grant, "A Generalized Adaptive Neural Network Fuzzy Inference Structure for Nonlinear Control", Proc. of the 14th ISCA International Conference on Computers in Industry and Engineering, Las Vegas, 2001.

[3] H.K. Lee and G. Lee, "Convergence Properties of Genetic Algorithms", Proc. of the ISCA Int'l Conference on Computers and Their Applications, Seattle, 2004.

[4] S. Nolfi and D. Floreano, Evolutionary Robotics, The Biology, Intelligence, and Technology of Self-Organizing Machines, MIT Press/Branford Books, Cambridge, 2000.

[5] K. Rajapakse, K. Furuta, and S. Kondo, "Evolutionary Learning of Fuzzy Logic Controllers and Their Adaptation Through Perpetual Evolution, IEEE Trans. On Fuzzy Systems, Vol. 10, No. 3, June 2002.

[6] W. Pedrycz, Computational Intelligence: An Introduction, CRC Press, 1997.

[7] P. Tang and G. Lee, "An Adaptive Fitness Function for Evolutionary Algorithms using Heuristics and Prediction", Proc. of the World Automation Congress, ISSCI, Budapest, 2006.

[8] G. Lee, G. and E. Grant, "Selection of the Generalized Adaptive Neural Network Fuzzy Inference Controller Parameters Using Evolutionary Simulated Annealing", Proc. of the ISCA Int'l Conference on Computer Applications in Industry and Engineering, Honolulu, 2008.

[9] S. Bhat, and G. Lee, "A Fuzzy Inference Function for Evolutionary Learning Systems", Proc. of the World Automation Congress, ISSCI, Seville, 2004.

[10] J. Kothari, E. Grant, and G. Lee, "Tuning an Adaptive Neural Network Fuzzy Inference Controller using Evolutionary Learning", Proc. Of the ISCA Int'l Conference on Computer Applications in Industry and Engineering, Orlando, 2004.

[11] P. Tang, S. Bhat, J. Kothari, and G. Lee, "A Modified Evolutionary Algorithm using a Heuristic Evaluation Function with Mutation Rate and Crossover Rate Tuning", Proc. Of the ISCA Int'l Conference on Computers and Their Applications, New Orleans, 2005.

[12] G. Lee and E. Grant, "On Parameter Selection for an Adaptive Fuzzy Neural Network Inference Controller using Evolutionary Tuning", Proc. of the ISCA Int'l Conference on Computers and Their Applications, Seattle, 2006.

[13] G. Lee, "On the Use of Hamming Distance Tuning for the Generalized Adaptive Neural Network Fuzzy Inference Controller With Evolutionary Simulated Annealing", Proc. of the IEEE Information Re-Use and Integration Conference, Las Vegas, 2011.

[14] G. Lee and E. Grant, "Adaptive Fuzzy Inference for Edge Detection Using Compander Functions", Proc. Of the ISCA Int'l Conference on Computers and Their Applications, Honolulu, 2007.

[15] J.J.E. Slotine, Tracking Control of Nonlinear Systems using Sliding Surfaces", Ph.D. Dissertation, Massachusetts Institute of Technology, 1983.