A Hierarchical Neuro-Fuzzy MRAC of a Robot in Flexible Manufacturing Environment

Kasim Al-Aubidy and Mohammed Ali
Computer Engineering Department, Philadelphia University, Jordan

Abstract: *In one hand, the Model Reference Adaptive Control (MRAC) architecture has been widely used in linear adaptive control field. The control objective is to adjust the control signal in a stable manner so that the plant's output asymptotically tracks the reference model's output. The performance will depend on the choice of a suitable reference model and the derivation of an appropriate learning scheme. While in the other hand, clusters analysis has been employed for many years in the field of pattern recognition and image processing. To be used in control the aim is being to find natural groupings among a set of collected data. The mean-tracking clustering algorithm is going to be used in order to extract the input-output pattern of rules from applying the suggested control scheme. These rules will be learnt later using the widely used Multi-layer perceptron neural network to gain all the benefits offered by those nets. A hierarchical neuro-fuzzy MRAC is suggested to control robots in a flexible manufacturing system. This proposed controller will be judged for different simulated cases of study to demonstrate its capability in dealing with such a system.*

Keywords: *MRAC, mean-tracking clustering algorithm, MLP neural nets, computer control, real-time systems, robots, FMS.*

Received July 29, 2003; accepted March 8, 2004

1. Introduction

Recently, low cost, small and middle production is made possible by a Flexible Manufacturing System (FMS). Flexible manufacturing systems represent efficiently grouped machine tools linked together for batch processing. The FMS consists of work cells, each cell is responsible of producing a group of parts with similar production processes [1, 16]. FMS is designed to accept raw materials at its input and automatically processes these raw materials into a certain product, which will be delivered at its output. The manufacturing process of these materials may take place on different work cells. Hence the capability and throughput of these systems are affected by the efficiency of the robots that move the product to and from these work cells. Moreover, inside each cell several machines may share to complete the manufacturing process [2]. In this case, the robot will play an important role in delivery, disposal and transport systems between cells and machines inside each cell.

The dynamic equations of the robot are a set of highly nonlinear differential equations. Therefore, the movement of the end effecter in a particular trajectory requires an efficient controller, which generates control signals applied by the robot joint actuators. There are many control strategies that can be applied to control robot joints. The traditional controllers cannot effectively control the motion of the robot. A controller based on the theory of the nonlinear control is suitable for the robot control [6]. Unfortunately, such controllers are not suitable for real-time FMS applications. This leads to think about controllers with intelligent capabilities to control the robot's operations in uncertain environments.

There are several types of control algorithms that can be used for joint control of the robot. Some of these use classical controllers, such as Proportional-Integral-Derivative (PID) [7] and adaptive controllers [10, 13], others use intelligent controllers based on neural nets [5, 8] and/or fuzzy logic [11, 15]. Conventional PID controller is still widely used in robotics. The performance of such a controller is not optimal and its parameters require readjustment, since the joint parameters are varying with time. Several tuning methods [17] have been published to obtain the controller parameters; however, most of these methods require the mathematical model of the robot. The nonlinear dynamic interactions of the robot joints are effectively minimized by applying sliding mode controllers [14]. Such a controller requires prior information about the robot parameters.

This paper deals with design and implementation of a neuro-fuzzy controller extracted from a model reference adaptive controller. The resultant functional controller is built based on the rules derived from applying a certain correcting formula to drift the system to behave as close as possible to the selected model. This can be applied to any joint in the manufacturing system, which represents the control activity in hierarchical control in order to make it suitable for real-time applications. The proposed

controller will improve the system performance by distributing the control tasks on multilevels.

2. Hierarchical Architecture

In our previous work [1, 2], the design and implementation of a hierarchical rout planner for FMS were proposed, as illustrated in Figure 1. The aim of the FMS rout planner is to obtain the optimal manufacturing routes for jobs according to well-designed cost function. The sequencing and monitoring module will monitor the competitive jobs to use the manufacturing cells and the required machines and robots. Also, this module can reveal the abnormal conditions in the system and generates feedback signals to the route planner to modify the old manufacturing routes to avoid the problems that may occur.

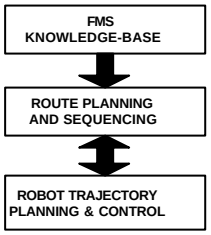


Figure 1. System organization.

For an FMS, several tasks of measurement, control, planning, operator communications, etc. can be distributed among a number of computers linked together and configured in a hierarchical structure. For the proposed system, given in Figure 2, five levels are recommended, these are:

- Measurement and actuation level: Provides on-line measurement and actuation database for the whole system.
- Control calculations level: Generates the required control signal for each joint.
- Controller parameters tuning level: Updates the controller parameters according to the actual behavior of the joint and the required trajectory.
- Robot trajectory planning level: Determines the input commands for each joint according to the robot trajectory.
- Planning and sequencing level: Obtains the optimal manufacturing routs for jobs, and then selects the required manufacturing cells, machines and robots.

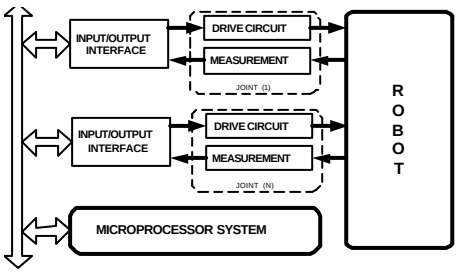


Figure 2. Hierarchical architecture.

This archi

- Information abstraction.
- Balancing precision with complexity.
- Multiple time scale operations.

It is assumed that the higher levels in the hierarchy, that is planning and sequencing, deal with a more abstract view of the control problem and do so in less precise terms. Moreover, the action-taking place at the higher levels affects the behavior of the system over a longer time span whereas the lower levels in the hierarchy operate on a faster time scale.

Figure 3 outlines the general layout of a robot system. It consists of a manipulator, and an input/output interface. A feedback interface is required to convert the position sensor signal of each joint into a digital code. The actuating signal generated by the control algorithm is loaded to the actuator of each joint through a feedforward interface.

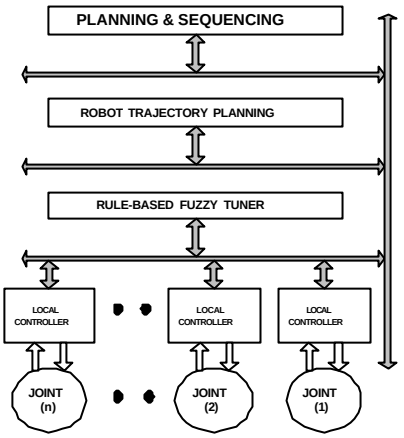


Figure 3. Robot system layout.

3. Model Reference Adaptive Control

The MRAC architecture has been widely used in the linear adaptive control field as shown in Figure 4 The control objective is to adjust the control signal in a stable manner so that the plant's output $y(t)$, asymptotically tracks the reference model's output $y_m(t)$. The performance of this algorithm depends on the choice of a suitable reference model and the derivation of an appropriate learning mechanism. Researchers in the sixties found that simple gradient-based learning rules were sometimes insufficient and there is no reason why this should not also be the case for more general nonlinear plant models and controllers [3, 4, 12].

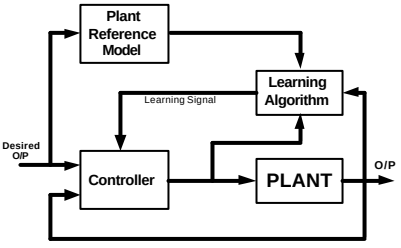


Figure 4. Model reference control architecture.

3.1. Control Strategy

The main function of the learning algorithm is to obtain the correct control signal *(ud)* corresponding to the desired output (y_d). The difference between the desired response (y_d) and the measured process output *(y)* is called the learning error *(eL)*. It is expected that this error will asymptotically approach zero, or a predefined small region, with increasing number of trials. The proposed learning scheme is shown in Figure 5.

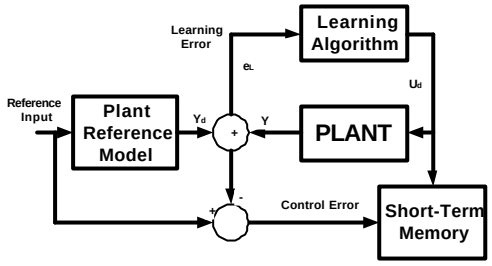


Figure 5. Block diagram of the proposed MRAC system.

3.2. Learning Algorithm

The object of the learning control is to determine the control input $u_d(t)$ by repetitive trial such that the error asymptotically tends to zero, or a prespecified small value, in the time interval of interest. The following error and derivative correction learning algorithm is proposed:

$$U_{k+1}(t) = u_k(t) + Pe_{Lk}(t+?) + Qe_{Lk}(t+?) \quad (1)$$

Where k denotes the instant number, ? is the time advances and p, q are learning gains for error signal and its derivative respectively.

It is noted that the error between the step command signal and the controlled output *y(t)* cannot be used as a learning basis because such a learning objective (step output) is clearly unrealistic. Therefore, a reference model is introduced, which specifies an achievable performance one would like to attain. Then, the learning error is used as a learning signal; see Figure 5. Equation (1) is used throughout the simulation for Single Input Single Output (SISO), P and Q are just scalar gains.

The main bottlenecks of this algorithm reside in choosing a suitable reference model and the time-consuming trail and error procedure in finding the suitable settings of the learning gains. The complete derivations of the previously discussed learning control algorithms are given by Linkens *et al.* [9].

4. NeuroFuzzy Controller Design

4.1. Mean-Tracking Algorithm

Clusters analysis has been employed for many years in the field of pattern recognition and image processing, the aim is being to find natural groupings among a set of collected data. A main problem always is the question of how many clusters there should be within a set of collected data. In practice, however, the number of clusters is problem dependent. The mean-tracking clustering algorithm was derived with the intention of dealing efficiently with operating data collected from high speed production machinery, the data is in the form of variable information taken from sensors in the plant, i.e.; variables such as speed, tension, temperature. In this case data is plotted in an n-dimensional space, each data point in the space corresponds to the machine state at a particular instant of time. Natural clusters of data points are then formed; all of the points within a cluster depict similar operating conditions to other points within that cluster. The center of gravity of the search data is then found by finding the mean value of all the points, which lie within the same cluster [18], see Figure 6 for description. In the proposed functional neuro-fuzzy controller the controlled data, input-output variables, are collected and clustered based on fuzzy-number using mean-tracking algorithm. The choice of the clusters will be clarified in simulation results section specifically in equation (9).

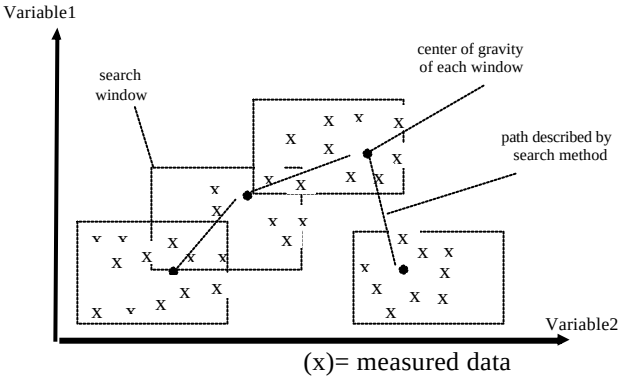


Figure 6. The mean-tracking cluster search method.

4.2. Controller Implementation

Depending on the methods for converting qualitative/ linguistic labels into quantitative/ numerical values, the structures of the resulting controllers are significantly different. Three possible controller input modes can be defined as in the following vectors:

$$\overline{Z} = [e_{m1}\ ce_{m1}, e_{m2}\ ce_{m2}, \ldots, e_{mT}\ ce_{mT}] \quad (2)$$

$$\overline{Z} = [e_{m1}\ se_{m1}, e_{m2}\ se_{m2}, \ldots, e_{mT}\ se_{mT}] \quad (3)$$

$$\overline{Z} = [e_{m1}\ ce_{m1}\ se_{m1}, \ldots, e_{mT}\ ce_{mT}\ se_{mT}] \quad (4)$$

where e_m, ce_m, and se_m are the measured error, change of error and sum of error. The three input types determined by the above representations (2, 3, 4) are called EC, ES, and ECS respectively. It is noted that they are analogous to classical PD, PI, and PID controllers respectively.

By explicitly embedding the meaning of the linguistic labels, the control j^{th} rule can be written as:

If e_m *is* $[C_j(e_m), d_j(e_m)]$ *and* e_m *is* $[C_j(ce_m), d_j(ce_m)]$ *Then* u *is* $[C_j(u), d_j(u)]$ (5)

where $d_j(e_m)$, $d_j(ce_m)$ are the input width for error and change-in-error respectively with centers $C_j(e_m)$ and $C_j(ce_m)$, while $d_j(u)$ is the width of the control action with center $C_j(u)$.

Now, it is possible to teach a neural network (NN) with only the central value vectors, i.e; the previous j^{th} linguistic rules become

$$If\ C_j(e_m)\ and\ C_j(ce_m)\ Then\ C_j(u) \quad (6)$$

While leaving the width vectors implicitly treated. One may ask how the fuzzy concept is handled in such paradigm. The answer as concluded before is that the NN inherently possesses some fuzziness, which is exhibited in the form of interpolation over new situations.

The clustering criterion based on mean-tracking algorithm using fuzzy number can be done over the vectors defined by equations (2, 3, 4) along with their corresponding control action (u) to get the centers of those variables to be learned using Back-Propagation Neural Network (BNN) of a structure shown in Figure 7. An ECS controller mode of clustered training vectors, conducted from controlling a process using the MRAC system are used to learn BNN of Figure 7.

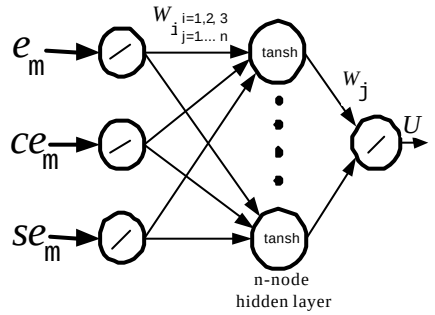


Figure 7. ECS type BNN represents the proposed functional neuro controller.

The functional neuro-fuzzy controller of Figure 7 has many advantages over that of MRAC. These can be summarized as follows:

- The size of the controller architecture decreases dramatically to offer less storage memory, less computations time and less fuzziness. Thus more efficient controller is available also a less cost hardware controller, if needed, can be easily achieved.
- More robust controller is obtained as will be verified in the simulation results.

5. Simulation Results

A robot model of an open loop type (0) second order transfer function:

$$G(s) = 1/s^2 + 7.5s + 0.09 \quad (7)$$

is taken to be control using the proposed control scheme. The closed loop response to a unit step change in input shows the sluggish over damped behavior of the system since it has low gain with a high damping ratio; moreover steady state error is detected. Thus the need is raised to include the effect of proportional, derivative and integral actions (ECS) type.

A model reference of:

$$Gm(s) = 5.4/s^2 + 5.4s + 5.4 \quad (8)$$

is chosen, after many trials, so that the robot system dynamic can follow such model with an applicable control action values. Applying equation (1) with gains p= 1.0 and q= 0.0, the required data for unit step change in input with sampling time of 0.1 sec. have been collected in the short-term memory to be used later.

Using these input-output data, 17-extracted rules are obtained as shown in Table 1, using the following clusters:

$$e_m = 0.0\ to\ 1.0\ step\ 0.15$$
$$ce_m = -0.85\ to\ 0.0\ step\ 0.15$$
$$se_m = 1.0\ to\ 9.5\ step\ 1.5 \quad (9)$$

Table 1. ECS clustered training vector of 17 rules used to train the BNN of Figure 7.

Center of e_m	Center of ce_m	Center of se_m	Center of Control Action
1	0	1	6.484656
1	-0.25	2.5	6.291598
1	-0.55	2.5	6.021482
0.9	-0.85	4	5.814579
0.75	-0.85	4	5.65887
0.6	-0.85	5.5	4.701158
0.45	-0.7	7	3.688067
0.45	-0.55	7	2.936129
0.3	-0.55	7	2.737491
0.3	-0.4	7	2.235832
0.3	-0.4	8.5	2.091233
0.15	-0.4	8.5	1.696041
0.15	-0.25	8.5	1.271224
0.15	-0.1	8.5	0.805575
0	-0.1	8.5	0.382604
0	-0.1	9.5	0.116563
0	0	9.5	0.090025

It is important to state here that since the integral action has been included by the accumulation in control signal, thus the sum of error will be of no use and it is added just to keep the notation of its existence.

Generally NNs can be used to memorize or discover the control strategy. Since the control law is extracted based on the MRAC scheme shown if Figure 5, thus BNN is applied here to memorize such control rules.

Applying the error back-propagation learning algorithm to BNN with the following characteristics:

Topology: 3-node input layer, 12-node tansh nonlinear hidden layer, 1-node linear output layer.

Parameters setting: Random initial weights of values between -0.5 to 0.5, steepness= 1.0, threshold= 1.0, learning rate= 0.1, momentum term= 0.0 and a stopping criterion of 0.01.

The 17 ESC fuzzy clustered rules shown in Table 1 are used to train the BNN illustrated by Figure 7. Convergence has been reached after 1645 iterations to

give final weights set which is inherently representing the controller behavior.

Figure 8 illustrates the two controlled responses of that conventional PID and neuro controllers along with that of uncontrolled one. The superiority of the neuro controller can be detected directly.

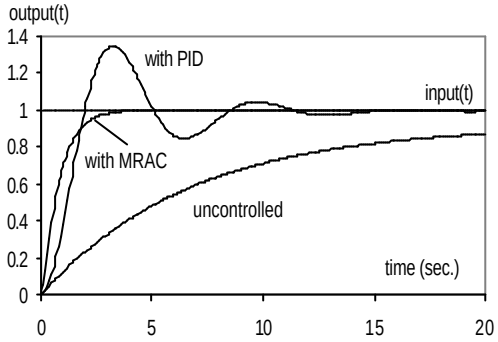


Figure 8. Comparisons between responses.

Many simulation tests have been achieved as well to verify the proposed controller capabilities as below:

Robustness test: Applying a disturbance of 20% of input value at steady state will not drift the controlled response into instability but to a slight acceptable steady state error of 0.037 value, which is within the tolerance band as shown in Figure 9. While Figure 10 illustrates the stand still controlled response if a time delay of 1 sec is occurred initially.

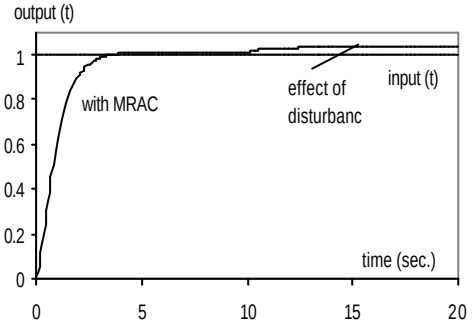


Figure 9. Effect of disturbance on the controlled response.

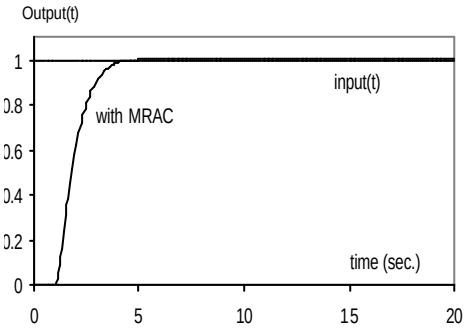


Figure10. Effect of time delay on the controlled response.

Tracking ability: Although the controller is extracted based on the unit step change in input, the *generalization feature* offered by neural networks gives the advantage of the ability to follow another input signal successfully. This is clearly shown in Figures 11 and 12, which illustrate the good tracking ability to both square, and staircase waves respectively.

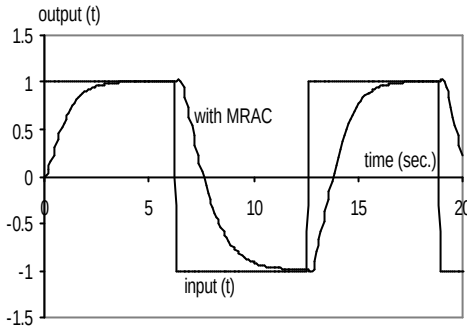


Figure 11. Controlled response tracking to a square wave input.

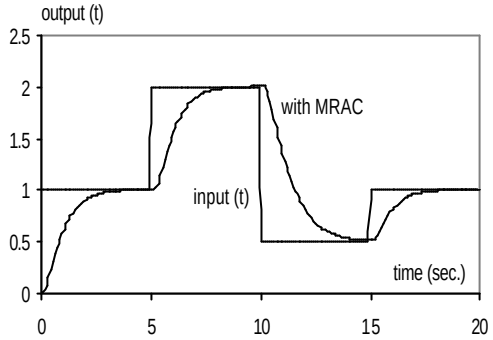


Figure 12. Controlled response tracking to a stair case input.

6. Conclusions

Many concluded points of high importance can be declared as follows:

- The complexity of MRAC is in choosing the appropriate model, which the underlying controlled system must follow.
- Suitable BNN parameters setting and topology are of high importance to gain fast convergence. Unfortunately there is no specified setting criterion, thus a trial and error procedure is applied.
- Representation of the input-output data achieved by using the mean-tracking algorithm is found to produce a robust functional neuro controller.
- The number of the extracted centers should be chosen neither large that gives a meaningless use of the clustering criterion nor so small that yields a bad representation of the original data.
- Generalization feature offered by neural networks gives the flexibility and adaptivity to use the resultant controller in many applications.
- A hardware form of the neuro controller can be easily achieved since it will be of small size and low cost.

References

[1] Al-Titinchi A. A. and Al-Aubidy M. K., "A Hierarchical Manufacturing Route Planner Design Based on Heuristic Algorithm: Design & Evaluation," *Systems Analysis Modelling Simulation Magazine*, vol. 42, no. 7, pp. 1119-1141, July 2002.

[2] Al-Titinchi A. A. and Al-Aubidy M. K., "Modeling an Interactive FMS Scheduler Using Colored Petri Nets," *in Proceeding of 2nd Middle East Conference on Simulation and Modeling (SCS-MESM'2000)*, Jordan, pp. 54-61, 2000.

[3] Astrom K. J. and Wittenmark B., *Adaptive Control*, Addison Wesley, Reading, MA, 1989.

[4] Brown M. and Harris C., *Neuro-Fuzzy Adaptive Modeling and Control*, Prentice-Hall International Ltd., UK, 1994.

[5] Ge S. S., Lee T. H., Gu D. L., and Woon L. C., "A One-Step Solution in Robotic Control System Design," *IEEE Robotics and Automation*, vol. 7, no. 3, pp. 42-54, 2000.

[6] Jamshidi M., Vadiee N., Rose J. T., and Ross T., *Hardware Applications of Fuzzy Logic Control, In Soft Computing: Fuzzy Logic, Neural Networks, and Distributed Artificial Intelligence*, in Aminzadeh F. and Jamshidi M. (Eds), Chapter 3, Prentice-Hall, USA, 1994.

[7] Kim S. W. and Lee J. J., "Neural Network Control by Learning the Inverse Dynamics of Uncertain Robotic System," *Journal of Institute of Control, Automatic System Engineering*, vol. 1, no. 2, pp. 88-93, 1995.

[8] Li Q., Poo A. N., Teo C. L., and Lim C. M., "Developing a Neurocompensator for the Adaptive Control of Robots," *in Proceedings of IEE, Control Theory Applications*, vol. 142, no. 6, pp. 562-568, 1995.

[9] Linkens D. A. and Nie J., "Constructing Rule-Bases for Multivariable Fuzzy Control by Self-Learning- Part 2," *International Journal of Systems SCI*, vol. 24, no. 1, pp. 129-157, 1993.

[10] Liu M., "An Adaptive Control Scheme for Robotic Manipulator," *in Proceedings of 5th International Symposium on Industrial Robotics*, USA, 1985.

[11] Palm R., "Fuzzy Controller for Sensor Guided Robot Manipulator," *Fuzzy Sets and Systems*, USA, 1989.

[12] Psalits D., Sideris A., and Yamamura A. A., "Neural Controllers," *in Proceedings of IEEE 1st International Conference on Neural Networks*, San Diego, CA, vol. 4, pp. 551-558, 1987.

[13] Spong W., *Robot Dynamics & Control*, John Wiley & Sons, USA, 1989.

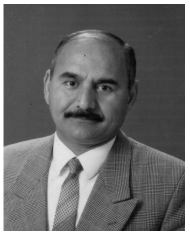
[14] Su C. Y. and Leung T. P., "A Sliding Mode Controller with Bound Estimation for Robot Manipulators," *IEEE Transactions Robotics & Automations*, vol. 9, no. 2, pp. 208-214, 1993.

[15] Tsai C. H., Wang C. H. , and Lin W. S., "Robust Fuzzy Model Following Control of Robot Manipulators," *IEEE Transactions Fuzzy Systems*, vol. 8, no. 4, pp. 462-469, 2000.

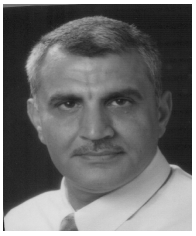
[16] Viswanadhan N. and Narahari Y., "*Performance Modeling of Automated Manufacturing Systems*," Prentice-Hall, USA, 1994.

[17] Voda A. A. and Landau I. D., "A Method for the Auto-Calibration of PID Controllers," *Automatica*, vol. 31, no. 1, pp. 41-55, 1995.

[18] Warwick K., "New Ideas in Fuzzy Clustering and Fuzzy Automata," *in Proceeding of the International ICSC Symposium on Fuzzy Logic, Swiss Federal Institute of Technology (ETH), Zurich, Switzerland*, May 26-27, 1995.



Kasim Al-Aubidy received his BSc and MSc degrees in control and computer engineering from the University of Technology, Iraq in 1979 and 1982, respectively, and the PhD degree in real-time computing from the University of Liverpool, England in 1989. He is currently an associative professor in the Department of Computer Engineering at Philadelphia University, Jordan. His research interests include fuzzy logic, neural networks, genetic algorithm and their real-time applications. He is the winner of Philadelphia Award for the best researcher in 2000. He is also a member of the editorial board of the Asian Journal of Information Technology. He has coauthored 3 books and published 47 papers on topics related to computer applications.



Mohammed Ali received his BSc, MSc and PhD degrees in computer and control engineering from the University of Technology, Iraq in 1981, 1993, and 1998, respectively. He is currently an assistant professor in the Department of Computer Engineering at Philadelphia University, Jordan. His research interests include fuzzy logic, neural networks, image processing, and computer interfacing. He has 6 published papers related to neurofuzzy and real-time computer control applications.