INTELLIGENT CONTROL ISSUES IN A DISTRIBUTED AUTONOMOUS ROBOTIC SYSTEM: *I-CUBES*

Mehmet Önder Efe[1], Lemi Daghan Acay[2], Cem Ünsal[3] and Pradeep K. Khosla[4]

[1]The Ohio State University, Electrical Engineering Department
Columbus OH 43210, U.S.A., {onderefe@ieee.org}
[2]Carnegie Mellon University, Institute for Complex Engineered Systems
Pittsburgh, PA, 15213-3890, U.S.A. {lacay@andrew.cmu.edu}
[3,4]Carnegie Mellon University, Electrical and Computer Engineering Department
Pittsburgh, PA, 15213-3890, U.S.A., {unsal@cs.cmu.edu}{pkk@ece.cmu.edu}

Abstract - The underlying idea in Distributed Autonomous Robotic Systems in general, and in self-reconfiguring modular robotic systems in particular concerns the coordination and cooperation of multiple objects to fulfill a specified task. However, the implementation phase typically faces two conflicting needs: Cheapest design and accuracy in fulfilling the task. In this paper, we describe a fuzzy control scheme for the servo systems of I-Cubes, which operate in a highly information-limited environment due to the size constraints. The results observed strongly recommend the approach due to its low computational cost together with the robustness against disturbances.

Keywords: Sliding Control, Fuzzy Control, Autonomous Robotics

I. INTRODUCTION

The driving force for autonomous robotic systems has been inspired from the biological counterparts, which can adapt themselves to fulfill a complicated task. Several examples can be visualized in the cell level biology of microorganisms. However, at a larger scale with man made machines, the concept of coordination and cooperation of individual cells depends heavily on the control of the interactions between the modules. At this stage, the low level control performance enters into the picture and plays a central role in achieving precise realization of the task. One challenge that constrains the design is the cost of the design. Since the design is subject to the individual control of a number of modules, and since the ultimate goal of the design is to obtain autonomous reconfiguration with low cost, it becomes inevitable to come up with robust control schemes that handle uncertainties with a highly information-limited environment. More explicitly, the controller for each motor must be capable of alleviating the drawbacks brought about by considerably low sampling rate, noise, low encoder resolution, unknown and nonlinear friction issues in gears and dead zone issues. Obviously, in such an environment, the adopted controller structure must be simple enough in order not to violate the computational requirements and must be versatile enough to fulfill the desired task.

In the literature, many architectures of intelligent control schemes have been discussed. Artificial neural networks, fuzzy inference systems and the ones utilizing genetic algorithms or the hybrid variants of these are just to name a few

[1]. In this paper, we utilize a fuzzy control scheme discussed in [2] in detail. The potential expectation has mainly been to exploit the information provided through the process of fuzzification, and then to utilize this partition based data in coming up with a crisp control decision.

This paper is organized as follows: The second section cites the recent contributions in the field of self-reconfigurable robotics and describes the capabilities of *I-Cubes* modules. The third section presents the control approach and the fourth section illustrates the experimental results obtained on the *I-Cubes* links. The conclusions are presented at the end of the paper.

II. I-CUBES

In this paper, we consider a class of self-reconfigurable robot links to demonstrate the control performance of the proposed approach. In the literature, a number of studies have demonstrated experimentally that the concept of self-reconfiguration could be attained autonomously through a self decomposition of the task into subtasks and executing a planning strategy based on the extracted decisions.

From a hardware point of view, the existing literature reports manual configuration [3], a modularly synthesized kinematics structure called *Tetrabot* [4], and cellular mobile robots with reconfiguration capability [5], *metamorphing hexagonal modules* [6], *self-repairing machines* [7], the *Cristalline robot* [8], which moves in a horizontal plane, and *Inchworm* [9], which moves in a vertical plane. The examples operating in 3D are *Polypod/Polybot* [10], *CONRO* [11], *robotic molecule* [11], self-reconfigurable structure [13], modular robot [14], *Proteo* [15] and *I-Cubes* [16]. A common property of what is presented in [10-16] is the capability of exploiting neighboring modules to fulfill the task.

I-Cubes system discussed previously in [16-19] is a bipartite robotic system composed of cubes forming a lattice and links (3-DOF manipulators) providing connections between the lattice elements. The components and the different CAD views of an *I-Cubes* link module are shown in Fig.1.

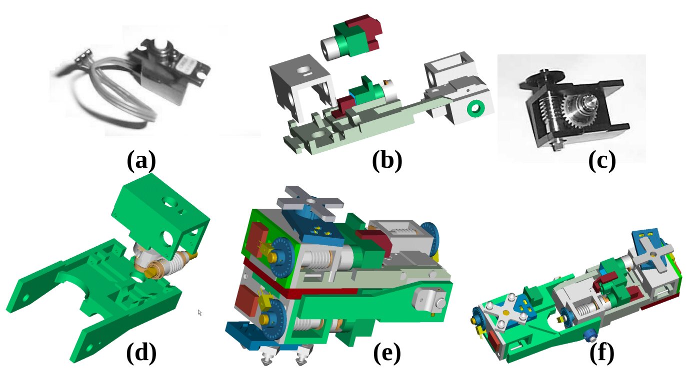

Fig. 1 The components and the physical allocation

In Fig. 1(a) the motor used for each joint is shown. The motor is Cirrus CS 21-BB Hi Performance Sub-Micro BB Servo. Its specifications are as follows: Size (cm) $2.1844\times 1.9812\times 1.0922$; Weight (gr) 9.0651; Speed (sec/60º) 0.09 under 4.8V, 0.078 under 6.0V; Torque (gr-cm) 1367.1387 under 4.8 V, 1654.9573 under 6.0 V. Fig. 1 (b) demonstrates how the motor is installed at the end of a single link. In Fig. 1 (c) and (d), the gearing that transfers rotary the motion of the motor to the link end is shown. A compact view is illustrated in Fig. 1(e) with encoders and two motors at the link ends and one motor in the middle of the whole system. The cross-shaped connectors at the link ends are designed for attachment and detachment and are not under investigation in this paper. The last subplot of Fig. 1 demonstrates the bottom view of a single link of *I-Cubes* system.

The block structure of the controlled system is depicted in Fig. 2, in which the motor is operating under an ordinary feedback loop.

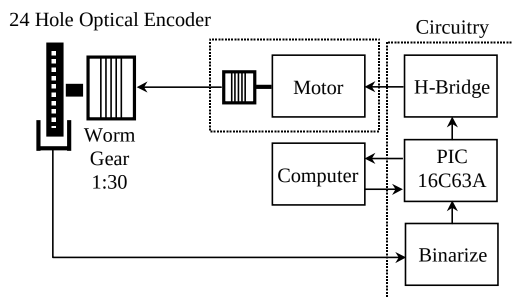

Fig. 2 Overall system structure

At each control instant, PIC sends 3 bytes to the computer. Two bytes of this information hold the measured position and the remaining byte is for the speed in terms of pulses and pulses/sec respectively. On the other hand the computer sends two bytes to the PIC. First byte is for the direction (or sign) and the other is for PWM (Pulse Width Modulation) duty-cycle. The above described process is achieved through a serial communication between the computer and the PIC. The two systems are synchronized at a rate of 160 msec (T), which is the adopted value of the control period. The process that takes place inside the computer include the interfacing, that has been designed on Java 2 platform, the control algorithm yet to be discussed, parameter tuning scheme and the serial communication software. The first apparent challenge here is the width of control period. The system will be forced to find the desired state through a set of observations that are sparse in time. One might suggest increasing the frequency, but the practical difficulty is that the optical encoder does not provide sufficiently accurate readings when the frequency is increased, and the observed positional data decreases to the order of a few pulses, which lead to an excessive noise when the speed is evaluated. On the other hand, changing the encoder and the relevant components with those providing better accuracy conflicts with the cost and size objectives of the *I-Cubes* system. In what follows, we describe the controller and the parameter tuning mechanism.

III. CONTROL STRATEGY

A. Controller

Contrary to what is postulated in the realm of predicate logic, representation of knowledge by fuzzy quantities can provide extensive degrees of freedom if the task to be achieved can better be expressed in words than in numbers. The concept of fuzzy logic in this sense can be viewed as a generalization of binary logic and refers to the manipulation of knowledge with sets, whose boundaries are unsharp [20]. Therefore the paradigm offers a possibility of designing intelligent controllers operating in an environment, in which the conditions are inextricably intertwined, subject to uncertainties and impreciseness.

Understanding the information content of fuzzy logic systems is based on the subjective judgements, intuitions and the experience of an expert. From this point of view, a suitable way of expressing the expert knowledge is the use of *IF antecedent THEN consequent* rules, which can easily evaluate the necessary action to be executed for the current state of the system under investigation.

Structurally, a fuzzy controller is comprised of five building blocks, namely, fuzzification, inference engine, knowledge base, rule base, and defuzzification. Since the philosophy of the fuzzy models is based on the representation of knowledge in fuzzy domain, the variables of interest are graded first. This grading is performed through the evaluation of membership values of each input variable in terms of several class definitions. According to the definition of a membership function, how the degree of confidence changes over the domain of interest is characterized. This grading procedure is called fuzzification. In the knowledge base, the parameters of membership functions are stored. Rule base contains the cases likely to happen, and the corresponding actions for those cases through linguistic descriptions, i.e. the IF-THEN statements. The inference engine emulates the expert's decision making in interpreting and applying knowledge about how the best fulfillment of the task is achieved. Finally, the defuzzifier converts the fuzzy decisions back onto the crisp domain [21].

The architecture that has been proposed by Wang [2] uses algebraic product operator for the aggregation of the rule premises and bell-shaped membership functions denoted by μ. The overall representation of this structure is given in (1), in which R and m stand for the number of rules contained in the rule base and the number of inputs of the structure.

$$\tau = \sum_{i=1}^{R} f_{ik} \left(\frac{\prod_{j=1}^{m} \mu_{ij}\left(u_{jk}\right)}{\sum_{i=1}^{R} \prod_{j=1}^{m} \mu_{ij}\left(u_{jk}\right)} \right) = \sum_{i=1}^{R} f_{ik} \Omega_{ik} \qquad (1)$$

with i^{th} rule as: IF u_1 is U_{1i} AND u_2 is U_{2i} AND … AND u_m is U_{mi} THEN $f_i=\phi_i$. In the IF part of this representation, the lowercase variables denote the inputs and the uppercase variables stand for the fuzzy sets corresponding to the domain of each linguistic label. The THEN part is comprised of the prescribed decision in the form of a scalar number denoted by ϕ_i. Clearly, the adjustable parameters of the structure are comprised of the parameters of the membership functions together with the defuzzifier parameters ϕ_i. Another common feature of the representation in (1) is the linearity of the output in the defuzzifier parameters.

B. Tuning Mechanism for the Motors of an I-Cubes Link

Although the differential representation of a DC servo we use in this project has ideally the form $\ddot{\theta} = -a_1\dot{\theta} + a_2\tau$ with θ and τ being the angular position and the input voltage respectively, the behavior on the presented mechanical design includes nonlinear friction, dead zone effect and a tiny backlash effect. It is rather tedious to construct a dynamic model containing the effect of such nonlinearities together with the mentioned difficulties in measurements. A suitable way of alleviating (or minimizing) the adverse effects of these, we utilize a tuning mechanism that actively changes the defuzzifier parameters ($\underline{\phi}$) of the fuzzy controller such that the motor dynamics is forced towards a sliding regime. For this purpose, set m=2, R=9, adopt the partitioning depicted in Fig. 3, and define the following quantities:

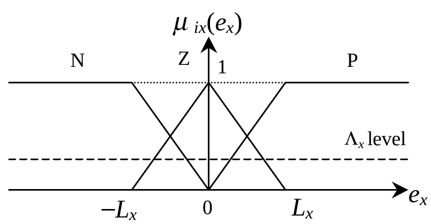

Fig. 3 Fuzzy partitioning of the input space for a generic parameter *x*

$$e_k = \theta_{dk} - \theta_k \text{ and } \dot{e}_k \cong (e_k - e_{k-1})/T = \Delta e_k/T \tag{2}$$

where a quantity with subscript k indicates its value at time kT, and θ_d stands for the desired angular position. For the quantization of input variables, we set L_e=5π/36 and $L_{\Delta e/T}$=π/36. Define the sliding line and the desired reaching law as below:

$$s_k = \dot{e}_k + \lambda e_k \text{ , } \lambda>0 \text{ and } s_{k+1} = (1-\lambda_1 T)s_k - \lambda_2 T \operatorname{sgn}(s_k) \tag{3}$$

with $\lambda_{1,2}>0$, $\lambda_1<T^{-1}$. In the tests, we used λ=0.3, λ_1=5.9375, λ_2=1.

Definition 3.1. The error on the applied control signal is described as $s_{Ck}=\tau_{dk}-\tau_k$ and is calculated as given in (4) [22-23].

$$s_{Ck} = s_k - (1-\lambda_1 T)s_{k-1} + \lambda_2 T \operatorname{sgn}(s_{k-1}) \tag{4}$$

Theorem 3.2. The adaptation of the defuzzifier parameters of the controller as given in (5) forces the error on the applied control signal (s_{Ck}) to zero level along with the sliding line.

$$\underline{\phi}_{k+1} = \underline{\phi}_k + \gamma \frac{\underline{\Omega}_k}{\underline{\Omega}_k^T \underline{\Omega}_k} \operatorname{sgn}(s_{Ck}) \quad (5)$$

where, γ is the uncertainty bound parameter. The proof of the theorem is presented in [22-24].

Theorem 3.3. If the discrete time sliding mode control task satisfies the following condition in the phase space, the stability condition $s_{Ck}(s_{Ck+1} - s_{Ck}) < 0$ is satisfied.

$$|\mu_{i1}(e_{k+1}) - \mu_{i1}(e_k)| \le 1 - \Lambda_e \text{ and } |\mu_{i2}(\dot{e}_{k+1}) - \mu_{i2}(\dot{e}_k)| \le 1 - \Lambda_{\dot{e}} \quad (6)$$

The conditions given in (6) with $0<\Lambda_x<1$ constrains the behavior of the jumps in the phase space partitioned by the rules as illustrated in Fig. 3. When the entire information of possible transitions between fuzzy regions are considered, it can be shown that that stability condition can be met with the described error measure and the tuning mechanism in (4) and (5) respectively. A detailed analysis and discussion on this issue is presented in [22-23].

IV. EXPERIMENTATION AND OBSERVATIONS

The first issue in the implementation stage is the time sparseness of the observed data and the necessity of filtering. For this purpose, we filter the velocity error before applying it to the controller and the output voltage before driving the motor. The filter structures for the velocity error and the control input are described in (7) and (8) respectively.

$$\hat{\dot{e}}_{k+1} = 0.5\hat{\dot{e}}_k + 0.5\dot{e}_k,\ \hat{\dot{e}}_0 = 0 \quad (7)$$

$$\hat{\tau}_{k+1} = 0.5\hat{\tau}_k + 0.5\tau_k,\ \hat{\tau}_0 = 0 \quad (8)$$

The coefficients seen above have been set by trial and error. Initially, the motor is at rest, i.e. at zero angle and at zero angular velocity. Since most of the tasks in the real-time implementation of *I-Cubes* entity is to move the motors on each link $\pi/2$ radians at a time, we test the control algorithm for a trajectory that is going to be used in multi-module *I-Cubes* entity. In Fig. 4(a) the position readings are illustrated. Clearly, the initial error is $\pi/2$ and has a tendency to converge to zero level. However, the numeric differentiation of the error signal carries some amount of uncertainty, which is illustrated in Fig. 4(b). In order to reduce the adverse effect of the near origin activity, we used the filter in (7) for velocity error and obtained the signal illustrated in Fig. 4(c). Although the filtered version contains such spikes as well, the magnitudes are comparable. The behavior observed in the phase space is depicted in Fig 4(d). The error vector starts moving towards the sliding line, which has the slope -0.3, and the control strategy forces the error vector to remain on the sliding regime. The first hit occurs at a point sufficiently far from the origin. However, the sliding regime starts after the second hit and then the error vector moves towards the origin with the associated well-known chattering phenomenon.

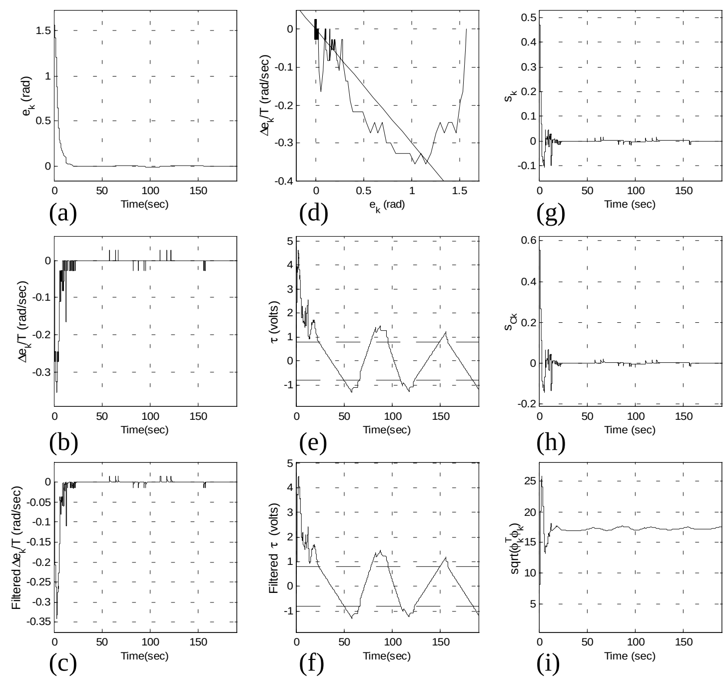

Fig. 4 Real-time observations from the control system

In the (e) and (f) subplots of Fig. 4, the calculated torque and the applied (or filtered) input signals are depicted with the same axial limits. It can easily be seen that the input voltage required to overcome the friction is approximately around 0.8 volts, below which the motor does not move at all. The dashed lines demonstrate the region in which the applied values do not result in any movement. Therefore, if there happens a very tiny tracking error in the vicinity of these critical values, the adaptation mechanism tries to overcome the frictional forces by tuning the parameters as fast as possible. The oscillation after approximately 20 seconds is because of this fact. The parameter γ for the tuning law of (5) has been set to 4. If this value is increased, the mobility of the learning is strengthened, whereas the small values lead to a slowly learning behavior and convergence takes relatively a long time. The smoothness of the applied control signal is another prominent feature of the proposed scheme.

In Fig. 4(g) and (h) the time evolution of the quantities given in (3) and (4) are depicted respectively. One particular observation here is the similarity of these two measures. The time evolution of the switching function of (3) more or less determines the behavior of the control error. A detailed discussion on this similarity can be found in [25], which suggests s_C as a monotonically increasing function of s for a class of systems in continuous time.

Lastly, the parameter evolution is depicted in Fig. 4(i). One suitable measure in evaluating the convergence issue in the adjustable parameter space is to examine the behavior of the quantity described below.

$$\mathrm{sqrt}\left(\underline{\phi}^{T}\underline{\phi}\right)=\sqrt{\sum_{i=1}^{R}\phi_i^2} \quad (9)$$

The trend of the evolution seen in this subplot suggests that the excited parameters are tuned until the goal is reached. During the real-time implementation, the initial values of the adjustable parameters have been set to zero. The result seen in Fig. 4(i) demonstrates that the tuning mechanism is internally stable, i.e. the parameters grow bounded and when the design specifications are met the tuning stops.

V. CONCLUSIONS

This paper discusses the issues related to real-time intelligent servo control mechanisms for a class of bipartite robotic systems called *I-Cubes*. Since one of the core issues in the design and implementation of the *I-Cubes* entity is to keep the modularity with cheap components, the control precision becomes a tedious objective due to the time sparse observations of data, numerical differentiation, noise, nonlinear friction effects, dead zone effects and limited mobility in applicable voltage range for the motors used. In order to address all these difficulties in an appropriate manner, we suggest the use of a fuzzy control scheme with triangular and stationary membership functions. The defuzzifier parameters of the controller are adjusted according to a rule that drives the servocontrol mechanism into a predefined sliding regime in discrete time. The results observed justify the practical difficulties of variable structure control scheme, however, these are alleviated through the use of an adaptation mechanism that can be activated or inactivated according to the path followed in the phase space. When the difficulties addressed are considered with what have been obtained, it can fairly be claimed that the proposed technique constitutes a good solution for precise control with cheap hardware.

The results presented throughout the paper have been obtained on a single servo operating with no load. Our scheduled work in this topic aims to demonstrate the feasibility and the efficacy of the proposed scheme for multiple servomechanisms together with the implementation of the controller on the PIC itself. Currently, the memory limitations of the used PIC processor do not allow us to transfer the control scheme into the PIC module.

ACKNOWLEDGMENTS

The work described in this manuscript was supported in part by Defense Advanced Research Projects Agency under contract DABT-63-97-1-0003, and by the Pennsylvania Infrastructure Technology Alliance, a partnership of Carnegie Mellon University, Lehigh University, and the Commonwealth of Pennsylvania's Department of Economic and Community Development.

REFERENCES

[1] J.-S. R. Jang, C.-T. Sun, E. Mizutani, *Neuro-Fuzzy and Soft Computing*, PTR Prentice-Hall, 1997.

[2] L. X. Wang, *A Course in Fuzzy Systems and Control*, PTR Prentice-Hall, 1997.

[3] C. J.-J. Paredis and P. K. Khosla, "Kinematic Design of Serial Link Manipulators from Task Specifications," *Int. J. of Robotics Research*, v.12, no.3, pp.274-287, 1993.

[4] B. Neville and A. Sanderson, "Tetrabot Family Tree: Modular Synthesis of Kinematic Structures for Parallel Robotics," Proc. IEEE/RSJ Int. Symp. of Robotics Research, pp.382-390, 1996.

[5] T. Fukuda and Y. Kawaguchi, "Cellular Robotic System as One of the Realization of Self-Organizing Intelligent Universal Manipulator," Proc. of the IEEE Int. Conf. on Robotics and Automation, pp.662-667, 1990.

[6] A. Pamecha, C.-J. Chiang, D. Stein and G. S. Chirikjian, "Design and Implementation of Metamorphic Robots," Proc. ASME Design Eng. Tech. Conf. and Comp. in Engineering Conf., Irvine CA, 1996.

[7] E. Yoshida, S. Murata, K. Tomita, H. Kurokawa and S. Kokaji, "Experiments of Self-Repairing Modular Machine," *Distributed Autonomous Robotic Systems* 3, H. Lueth, R. Dillmann, P. Dario, H. Wörn, (eds.), Springer-Verlag, pp.119-128, 1998.

[8] M. Vona and D. L. Rus, "A Physical Implementation of the Self-reconfiguring Cristalline Robot," Proc. of the IEEE Int. Conf. on Robotics and Automation, pp.1726-1733, 2000.

[9] K. Kotay and D. L. Rus, "The Inchworm Robot: A Multi-Functional System," *Autonomous Robots*, v.8, no.1, pp.53-69, 2000.

[10] M. Yim, D. Duff and K. D. Roufas, "PolyBot: A Modular Reconfirurable Robot," Prof. IEEE Int. Conf. on Robotics and Automation, pp.514-520, 2000.

[11] A. Castano, W.-M. Shen and P. Will, "CONRO: Towards Deployable Robots with Inter-Robots Metamorphic Capabilities," *Autonomous Robots*, v.8, no.3, pp.309-324, 2000.

[12] K. Kotay and D.L. Rus, "Motion Synthesis for the Self-Reconfiguring Robotic Molecule," Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, v.2, pp.843-851, 1998.

[13] S. Murata, H. Kurokawa, E. Yoshida, K. Tomita and S. Kokaji, "A 3-D Self-Reconfigurable Structure," Proc. IEEE Int. Conf. on Robotics and Automation, pp.432-439, 1998.

[14] E. Yoshida, S. Murata, A. Kaminura, K. Tomita, H. Korokawa and S. Kokaji, "Motion Planning of Self-reconfigurable Modular Robot," Proc. of the 7th Int. Symp. on Experimental Robotics, pp.375-384, 2000.

[15] H. Bojinov, A. Casal and T. Hogg, "Emergent Structures in Modular Self-reconfigurable Robots," Prof. IEEE Int. Conf. on Robotics and Automation, pp.1734-1742, 2000.

[16] C. Ünsal and P. K. Khosla, "A Multi-Layered Planner for Self-Reconfiguration of a Uniform Group of I-Cube Modules," to appear in Proc. 2001 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, 2001.

[17] C. Ünsal, H. Kiliççöte, M. Patton and P. K. Khosla, "Motion Planning for a Modular Self-reconfiguring Robotic System," Distributed Autonomous Robotic Systems 4, Springer-Verlag, pp.165-175, 2000.

[18] C. Ünsal, H. Kiliççöte, and P. K. Khosla, "A Modular Self-Reconfigurable Bipartite Robotic System: Implementation and Motion Planning," *Autonomous Robots*, v.10, no.1, pp. 23-40, 2001.

[19] C. Ünsal and P. K. Khosla, "Solutions for 3-D Self-reconfiguration in a Modular Robotic System: Implementation and Motion Planning," Proceedings of SPIE, Sensor Fusion and Decentralized Control in Robotic Systems III, November 2000.

[20] Yen, J. and R. Langari, *Fuzzy Logic*, PTR Prentice-Hall, New Jersey, 1999.

[21] Passino, K. M. and S. Yurkovich, *Fuzzy Control*, Addison-Wesley, California, 1998.

[22] M. O. Efe, "Design of a Fuzzy Variable Structure Control Strategy for Discrete Time MIMO Nonlinear Systems," *IEEE Trans. on Fuzzy Systems* (submitted for publication).

[23] M. O. Efe, "Fuzzy Variable Structure Control of a Class of Nonlinear Sampled-Data Systems," *Dynamics and Control* (submitted for publication)

[24] M.O. Efe, O. Kaynak and X. Yu, "Sliding Mode Control of a Three Degrees of Freedom Anthropoid Robot by Driving the Controller Parameters to an Equivalent Regime," *Trans. of the ASME: Journal of Dynamic Systems, Measurement and Control*, v.122, no.4, pp.632-640, December 2000.

[25] M.O. Efe, "Variable Structure Systems Theory Based Training Strategies for Computationally Intelligent Systems," Ph.D. Dissertation, Bogazici University, 2000.