also used to control parallel robots; for example, Yongsheng *et al.* [4] formulated the enhanced fuzzy sliding mode controller for a 3-DOF parallel manipulator. The method of neural networks was also attractive and used by researchers for control actuated parallel robots, e.g., Akbas [5], and Li *et al.* [6].

The PID controllers can be described by robust performances across a wide range of operating conditions and their functional simplicity. However, the high nonlinear nature of the parallel robot means that a PID controller can perform well only at a particular operating range**.** The PID controllers can be basically divided into two categories. Firstly, the PID parameters are fixed over the entire control process; however, it is difficult to obtain satisfactory performances when the control system is highly nonlinear and heavily coupled. Secondly, in self-tuning PID, the parameters can be manipulated online based on the parameters estimation [7]. In order to obtain global results, it is necessary to re-tune the PID controller when the operating range is changed, and different techniques from nonlinear control theory are required [8].

Robotic machines, unlike humans, lack the ability to solve problems using imprecise information where it requires restrictive assumptions for the plant model and for the control to be designed (e.g., linearity). To emulate this ability, fuzzy logic and fuzzy sets are introduced [8]. The fuzzy controller can be designed without knowing the mathematical model of the system, and instead mimics human operators' thinking processes through linguistic rules. These rules reflect human knowledge about how to control the dynamic system. In addition, unlike PIDs, fuzzy controllers are nonlinear and adaptive in nature, thereby giving a robust performance under parameter variations and load disturbance effect [7].

Although fuzzy logic systems, which can reason with imprecise information, are good at explaining their decisions, they cannot automatically acquire the rules used to make those decisions [9]. On the other hand, artificial networks are good at recognizing patterns and have ability to train the parameters of a control system, but they are not good at explaining how they reach their decisions. These limitations in both systems have stimulated the creation of intelligent hybrid systems (like neuro-fuzzy system) where the two techniques are combined in such a manner that the limitations of the individual techniques are overcome. The neuro-adaptive learning techniques provide a method for the fuzzy modeling procedure to acquire information about a data set. This technique gives the fuzzy logic capability to compute the membership function parameters that effectively allow the associated fuzzy inference system to track the given input and output data [9].

The ANFIS control algorithm is very attention due to its robustness for nonlinear systems. Adhyaru and Jimit [10], Bachir and Zoubir [11], and Ngo et al.[12] used the ANFIS to control serial robot by training the input/output PID control data. Under the conditions of uncertainly, a method to identify the model parameters of parallel manipulators is to use the ANFIS control algorithm. Such an algorithm can be performed in a real time control application [9].

This paper is mainly concerned with the applications of Fuzzy and ANFIS that are contained within the AI techniques to control a hydraulically driven parallel robot. In the second section, the dynamics of the parallel robot is analyzed considering that the rod and joints inside every rigid cylinder are flexible, while the cylinders and moving plate are rigid bodies. The floating frame of reference method is used to model flexible components using FEM and then used to assemble the dynamics of the parallel robot model through the application of Lagrange's equation and Lagrange multiplier method. In the third section, a hydraulic control system is designed and a PID control law is used. In section four, the fuzzy-PI self tuning of the gains (Kp and K_d) for each hydraulic cylinder controller is designed, and in section five, the ANFIS-PI self tuning also control is also designed for the same reason. Finally, using the real robot parameters, the simulation studies were conducted out to demonstrate the performance of the proposed controllers.

II. Dynamic Analysis of Parallel Robot

The floating frame of reference method (FFRF) can be applied to bodies that undergo large body translations and rotations as well as elastic deformations with respect to a frame of reference. The deformation of a flexible body (rod) with respect to its frame of reference can be formulated an Euler-Bernoulli beam and using a finite element method, where each rod is meshed into 6 elements and 7 nodes. Fig. 2 explains the floating frame of reference (FFR) coordinate systems used for describing the changes in the position of a point P^i in a deformed body i [1], [13].

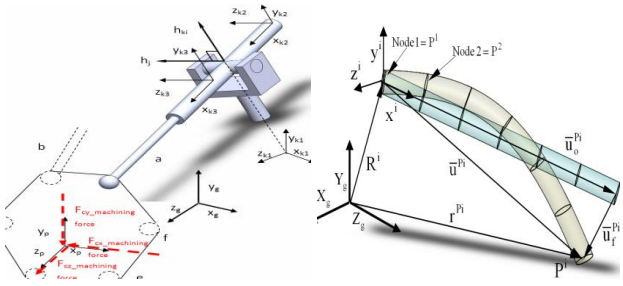


Figure 2. The position of node Pi in floating frame coordinates and finite elements of i^{th} flexible rod.

The dynamic nonlinear differential algebraic equation DAE that describes the overall rigid body motions of the moving platform and 6 rigid cylinders as well as the 6 flexible rods is [1], [13]:

$$\begin{bmatrix} I & 0 & 0 \\ 0 & M & C_q^T \\ 0 & C_q & 0 \end{bmatrix} \begin{bmatrix} \dot{q} \\ \ddot{q} \\ \lambda \end{bmatrix} = \begin{bmatrix} \dot{q} \\ Q^e + Q^v + Q^f \\ Q^c \end{bmatrix} \quad (1)$$

where M is the mass-inertia matrix; C_q, the Jacobian matrix of the nonlinear constraint equations; q, the vector of n generalized coordinates of all bodies of the parallel robot; λ, the vector of Lagrange multipliers; Q^e, the vector of generalized forces, and Q^v the quadratic velocity

vector. And Q^f =[0 0 $K^i q_f$]^T the vector of elastic forces; K, the diagonal orthonormalized modal stiffness matrix and q_f^i, the vector of elastic coordinates [13].

III. Derivation of Hydraulic Forces

The parallel robot is mainly driven by water hydraulic servo actuators for two reasons first: hydraulic systems can offer high power density, which permits lightweight constructions, and secondly, water hydraulics is clean and suitable for the environment inside the ITER vacuum vessel. However, the use of water hydraulic drive is a challenge because of the limit of the flow rate of the servo valve. The speed cannot be very high (over 3 m/min), since, otherwise, the speed error will be greater than acceptable and the robot cannot follow the track accurately [1]. The water hydraulic system is composed of six cylinders for the parallel robot, and each one is controlled by a Moog Type-30 servo valve. The pressures and flow rates in the system can be derived as follows.

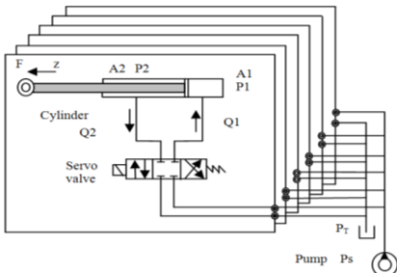


Figure 3. Hydraulic components of the parallel robot.

Assume that P_1, P_2 are the pressures of the infill and return water cavity, respectively (Pa); A_1, A_2, the effective action areas of the infill and the return water, respectively (m^2); and Q_1, Q_2, the flow of the infill and the return water, respectively, (m^3/s) (Fig.3). Then:

$$P_L = F/A_1 = P_1 - P_2(A_2/A_1) \quad (2)$$

$$Q_L = Q_1 + nQ_2/(1+n^2) = Q_1 \quad (3)$$

In this formula, n = A_2/A_1. The resultant force produced inside each cylinder (F) can be derived from the pressures acting on the piston as follows:

$$A_1p_1 - A_2p_2 = A_1p_L = m\ddot{x} + b\dot{x} + F \quad (4)$$

where x is the displacement of a piston (rod); b, the coefficient of friction (impedance coefficient, N.s/m), m is the mass of a rod. After some simplifications and substitutions, the following can be achieved:

$$Q_L = A_1\dot{x} - \frac{Vt}{2(1+n^2)Be}\dot{P}_L + C_{tc}P_L \quad (5)$$

here, Be is the water bulk modules; C_{ic} is the internal leakage. *Vt*: is total volume, and $C_{tc}=((1+n)/(1+n^3))C_{ic}$. The application of the Laplace transformation to (4) and (5) leads to the transfer function of the actuating unit of the valve controlled cylinder [1].

As shown in Fig.4, the trajectory generator calculates the rod (leg) position that is formed as a 6×1 (x_d) vector feeding the PID control input. The PID controller produces a 6×1 control vector, z, which should fed to the hydraulic servovalve to produce the leg force F applied to the prismatic joint actuators of the manipulator to produce 6×1 output vectors x, which include actual rod positions. These are fed back to the controller [1]. The errors between the predicted and presented rod positions are used to determine the required force of the six actuators as follows:

$$u(k+1) = x_d(k+1) - x(k) \quad (6)$$

where x_d is the desired local displacement of a rod.

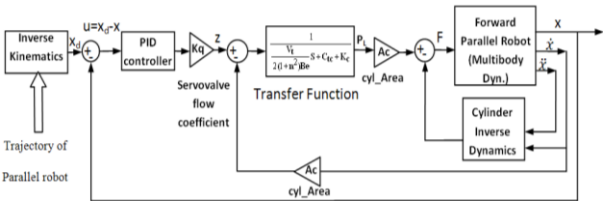


Figure 4. Block diagram of the control system of the parallel robot.

IV. Fuzzy-PID Self Tuning and Controller and Its Membership Function

The parallel robot is considered a six closed mechanical chains that make the dynamics of the parallel manipulators highly complex and their dynamic models highly nonlinear. Because of these conditions, a conventional PID controller cannot reach satisfactory results. According to Tian [7], a self-tuning parameter fuzzy PID controller provides a control of the system with excellent performance in reliability, stability, and accuracy. The basic approach is to try to detect inputs when the controller is not properly tuned and then seek to adjust the PID gains to improve the performance. The schematic structure of the self-tuning-parameter fuzzy PID controller is given in Fig. 5.

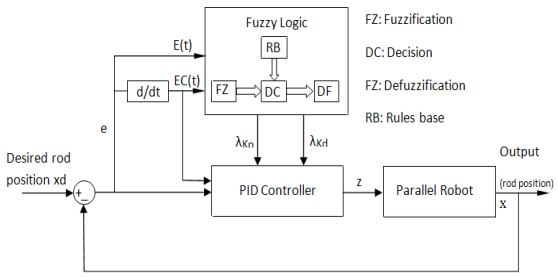


Figure 5. Self-tuning-parameter fuzzy PID controller structure.

The fuzzy controller is composed of the following four elements:1- *a fuzzification*, 2- *a rule-base or knowledge base* (a set of **If-Then** rules), 3- *an inference mechanism or decision making mechanism* (also called a "fuzzy inference" module), and 4- *a defuzzification*.

In Fig. 5, the input is the reference value of the rod position and the output is the actual rod position. Inputs for the fuzzy block are rod length error *E* and the time derivative of a rod length error, *EC*. The PID controller parameters K_p and K_d are self-tuned according to the following logic rules by a *fuzzy inference*.

After all the inputs and outputs are defined for the fuzzy controller, the fuzzy control system can be specified. The linguistic description is provided by a control "expert" on how to tune the PID parameters.

Next, the linguistic quantification above specifies a set of rules (a rule-base) that capture the expert's knowledge about how to control a rod position. The knowledge of the process, which is a fuzzy model, is always described using simple fuzzy linguistic rules instead of precise mathematical functions. The general expression of a rule for this control system is as follows:

R_i: IF E is NEm and EC is NECn, THEN λ_{Kp} is NK1, λ_{Kd} is NK2,

where λ_{Kp} and λ_{Kd} denote output of the FLC, denote the adjustment coefficients of the PID parameters, NEm Є NE, NECn Є NEC, NK1, NK2 Є NK.

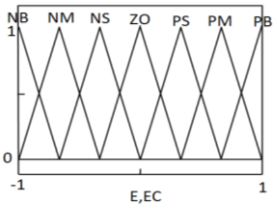


Figure 6. Membership functions of inputs

Fig. 6 shows two inputs: the rod position error *E* and the derivation of the error *EC*. Both *E* and *EC* are divided into seven values as {NB, NM, NS, ZO, PS, PM, PB}, where NB: Negative Big, NM: Negative Medium, NS: Negative Small, ZO: Zero, PS: Positive Small, PM: Positive Medium, PB: Positive Big.

The horizontal axis in Fig. 6 illustrates the scaling gain for a rod position error *E* and its time differential *EC*. Triangle membership functions were chosen as they are the most common and easy to implement in an embedded controller. Each of the triangles represents an area of the effect of rules. Similar interpretations of linguistic values were made in the definition of the membership functions on the outputs.

TABLE I. THE RULE BASE FOR THE TUNING OF THE CONTROL SYSTEM.

		Rod Position Error *E*						
		NB	NM	NS	AZ	PS	PM	PB
Differential of error *EC*	NB	VB/ ZE	B/V S	SB/ S	S/M B	SB/ SB	MB/ S	B/V S
	NM	B/V S	MB/ S	SB/ SB	S/S B	SB/ SB	MB/ S	MB/ VS
	NS	MB/ VS	SB/S B	S/S B	VS/ B	S/M B	SB/ SB	SB/S
	AZ	SB/S	S/M B	VS/ B	ZE/ VB	VS/ B	S/M B	SB/S
	PS	SB/S	SB/ MB	S/S B	VS/ B	S/S B	SB/ SB	MB/ VS
	PM	MB/ VS	MB/ S	SB/ SB	S/S B	SB/ SB	MB S	B/V S
	PB	B/V S	MB/ S	SB/ SB	S/S B	SB/ S	B/V S	VB/ ZE

By combining the fuzzy sets of inputs for the rod position error (7) and the differential of the rod position error (7), there are totally 7×7 = 49 rules for tuning one controller output. Because we have two outputs (Kp and K_d), there are 49 × 2 = 98 rules for each servo controller. The rule base for the tuning of the control system is shown in Table 1.

Table 1 illustrates what rule is effective when a specific combination of the rod position error *E* and its time differential *EC* is presented. For instance, the rod position error *E* is "located" in the NS triangle, and the differential of error *EC* in the ZO triangle. The combination of this information tells us that the output for K_p follows the NS triangle rule, and K_d follows the PS triangle rule.

The *inference process* or *decision* generally involves two steps: 1) The premises of all the rules are compared with the controller inputs to determine that rules apply to the current situation, 2) The conclusions (what control actions to take) are determined using the rules that have been determined to apply at the current time. The conclusions are characterized with a fuzzy set that represents the certainty that the input to the plant should take for various values [14].

The "AND" operator is applied, and then the membership degree of the output in a rule can be calculated as:

$$\mu(z) = \min \{\mu(x); \mu(y)\} \quad (7)$$

Based on the input information (*E, EC*), the triangle membership function is chosen. The result $\mu(z)$ should undergo a defuzzification process. *Defuzzification* refers to the way a crisp value is extracted from a fuzzy set as representative value, by combining the results of the inference process and then computing the "fuzzy centroid" of the area (of the chopped off triangles). The result is the $\lambda(E;EC)$ weight coefficient (crisp value) which depends on *E* and *EC* and is used for calculating K_p and K_d, which can be summarized as follows [15]:

$$\lambda^{COG} = \sum_{i=1}^{N} \mu(z_i).z_i \,/\, \sum_{i=1}^{N} \mu(z_i) \quad (8)$$

Now, the tuned parameters of the PID controller can be found as follows [7]:

$$K_p = K_{p,Min} + \lambda_{K_p}(E_L, EC_L)(K_{p,Max} - K_{p,Min}) \quad (9)$$

$$K_d = K_{d,Min} + \lambda_{K_d}(E_L, EC_L)(K_{d,Max} - K_{d,Min}) \quad (10)$$

where λ is a weight coefficient, and $K_{p,\max}$, $K_{p,\min}$ $K_{d,\max}$, and $K_{d,\min}$ are the maximum and minimum limits for the proportional gain and the integral gain, respectively. These limits were chosen from several tests for the conventional PID controller. Fig. 7 shows outputs λ_{Kp} and λ_{Kd}.

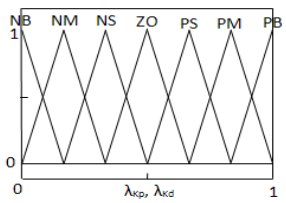


Figure 7. Membership functions of outputs.

V. ANFIS CONTROLLER FOR TUNING PID GAINS

The ANFIS discriminates itself from normal fuzzy logic systems by the adaptive parameters, i.e., both the premise and consequent parameters are adjustable. The most noteworthy feature of the ANFIS is its hybrid learning algorithm. The adaptation process of the parameters of the ANFIS is divided into two steps [11]. For the first step of the consequent parameters training, the Least Squares (LS) method is used because the output of the ANFIS is a linear combination of the consequent parameters. The premise parameters are fixed at this step. After the consequent parameters have been adjusted, the approximation error is backpropogated through every layer to update the premise parameters as the second step. This part of the adaptation procedure is based on the gradient descent principle, which is the same as in the training of the BP neural network. The consequence parameters identified by the LS method are optimal in the sense of least squares under the condition that the premise parameters are fixed [11].The training process stops whenever the designated epoch number is reached or the training error goal is achieved**.** A combination of such intelligent systems, like ANFIS provides even better results than just neural networks or fuzzy control [10].

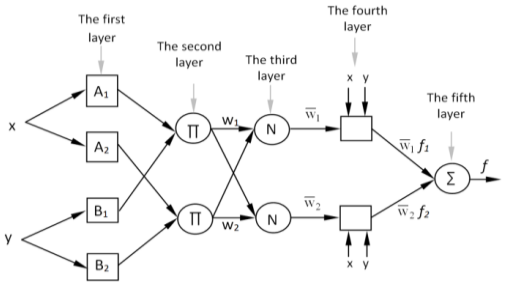


Figure 8. A typical architecture ANFIS structure.

To improve the reliability of the controller by the error minimization approach, and to overcome the awkward task of choosing membership functions of fuzzy controller, ANFIS are used in a parallel structure and embedded to the control system. In this implementation, error vector is computed for each ANFIS by using the difference between the actual rod positions generated by manipulator's dynamic model and the desired rod positions. After the off-line training, the output values of the gains Kp and Kd generated by twelve ANFIS are applied to the PID controller of each cylinder. The results are evaluated to select the network generating the best result. It is then assigned as the ANFIS controller for actual time steps. The architecture of the used ANFIS is shown in Fig. 9.

A typical architecture of used ANFIS is shown in Fig.8; here in which a circle indicates a fixed node, whereas a square indicates an adaptive node. For simplicity, we consider two inputs *x, y* and one output *f*. Among the many FIS models, the Sugeno fuzzy model is the most widely applied one for its high interpretability and computational efficiency, and built-in optimal and adaptive techniques. For each model, a common rule set with two fuzzy if-then rules can be expressed as [15]:

Rule i: if x(=e) is A_i and y(= $\dot{e}$) is B_i, then $f_i=p_ix + q_iy + r_i$

where A_i and B_i are fuzzy sets in the antecedent and z=f(e, $\dot{e}$) is a crisp function in the consequent.

The ANFIS controller generates continuous changes in the reference PID parameters *Kp* and *Kd*, based on the i[th] rod position error e and derivate of the error $\dot{e}$, error defined as: $e=x-x_d$, where x_d and x are the reference and the actual displacement of the i[th] rod of parallel robot, respectively. In this study, each ANFIS consists of five layers as follows [15]:

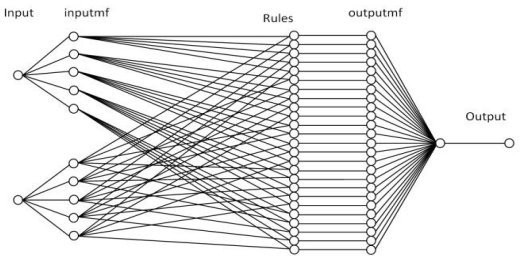


Figure 9. ANFIS structure for each cylinder controller.

Layer 1: In this layer, every node is adaptive and the output of each node i is the degree of membership of the input to the fuzzy membership function (MF) represented by the node, i=1,…,5. In this paper, the node function is a generalized bell membership function:

$$\begin{aligned} O_i^1 &= \mu_{Ai}(e=x) = \frac{1}{1+\left|\frac{x \quad c_i}{a_i}\right|^{2b_i}} \\ O_i^1 &= \mu_{Ai}(e),\ i=1,5 \\ O_i^1 &= \mu_{Bi\ \ 5}(\dot{e}),\ i=6,10 \end{aligned} \tag{11}$$

O_i^1 is the output of the i[th] node, A_i and B_i are the fuzzy sets in parameters form; x is the input to the node i. $\{a_i , b_i , c_i\}$ are premise parameters.

Layer 2: The total number of rules is 25 in this layer. Each node output represents the activation level of a rule:

$$O_i^2 = w_i = \mu_{Ai}(e)\,\mu_{Bi}(\dot{e}),\ \ i=1....,5 \tag{12}$$

Layer 3: Fixed node i in this layer calculates the ratio of the i[th] rules activation level (firing strengths) to the total of all activation level; this layer is called normalized firing strengths:

$$O_i^3 = \overline{w}_i = w_i \Big/ \sum_{i=1}^{n} w_i \tag{13}$$

Layer 4: Adaptive node i in this layer calculates the contribution of the i[th] rule towards the overall output, with the following node function:

$$O_i^4 = \overline{w}_i f_i = \overline{w}_i (p_i x + q_i y + r_i) = \overline{w}_i (p_i e + q_i \dot{e} + r_i) \tag{14}$$

where $\overline{w}_i$ is the output of layer 3, and $\{p_i , q_i , r_i\}$ are the consequent parameters.

Layer 5: The single node in this layer computes the overall output as the summation of all incoming signals, which is expressed as:

$$O_i^5 = \sum_{i=1}^{n} \overline{w}_i f_i = \sum_{i=1}^{n} w_i . f_i \Big/ \sum_{i=1}^{n} w_i \qquad (15)$$

The learning rule is the backpropogation gradient descendent, which calculates the error signals recursively from the output layer backward to the input nodes. The task of the learning algorithm for this architecture is to tune all the modifiable parameters to make the ANFIS output match the training data. The overall output is a linear combination of the modifiable parameters.

The training algorithm requires a training set defined between inputs and outputs [14]. The input and output pattern set have 50000 rows. Figure 10, a, b, c, d show optimized membership function for e and $\dot{e}$ after training for each cylinder controller. Figure 9 shows the ANFIS model structure. The number of epochs was 100 for training. The number of MFs for the input variable e and $\dot{e}$ is 5, respectively. The number of rules is then 25(5 × 5=25). The generalized bell (Cauchy) MF is used for each input variables. It is clear from (11) that the bell MF is specified by three parameters. Therefore, the ANFIS used here contains a total of 105 fitting parameters, of which 30 (5×3 + 5×3=30) are the premise parameters and 75(3×25=75) are the consequent parameters for each cylinder controller.

VI. Parallel Robot Parameters, Simulation Results and Discussion

The actual values of the system parameters of the parallel robot of each element are tabulated in Table 2. The local position of the universal joint in the base plate and the local position of the spherical joint in the moving plate are given below. The tip point of each rod is initially located at 0.35m from the cylinder outlet. Young's modulus is 2.07e11 N/m^2.

Base points (local): [0.1658 cos(120(1-i)+(90±14.851)) 0.1658 sin(120(1-i)+(90±14.851)) 0] i =1,2,3.

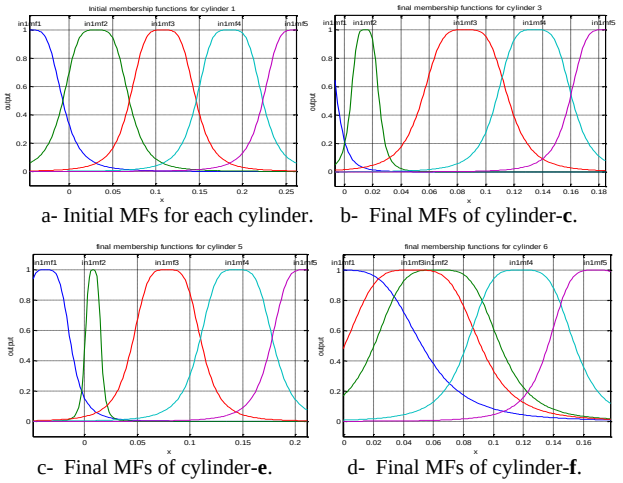
End-effector points (local): [0.1296 cos(120(1-i)+(90 ±45.485)) 0.1296sin(120(1-i)+(90±45.485)) 0] i =1,2,3.

TABLE II. Mass and Inertia Properties of Parallel Robot' Elements.

Element	Mass (kg)	Izz = Iyy (Kg m^2)	Ixx (Kg m^2)	Ixy (Kg m^2)
cylinder	4.589	0.2159864	2.89683e-3	0
rod piston	3.683	0.1371788	4.14392e-4	0
moving plate	28.92	0.1867	0.3622	0

The moving plate (end effector) is simulated to track a trajectory of: x= 0; y= -0.1t cos(42.87); z= +0.1t sin(42.87), with machining forces: Fx= 1000sin(2πft) N, Fy= 700sin(2πft) N, and Fz= -600sin(2πft) N, f=20 Hz. The equation of motion was solved using the Runge-Kutta method with the initial values of generalized elastic deformation $q_f^i = 0_{3x1}$, velocity $\dot{q}_f^i = 0_{3x1}$, and first three non-rigid body modes, and each rod is assumed a simply supported beam.

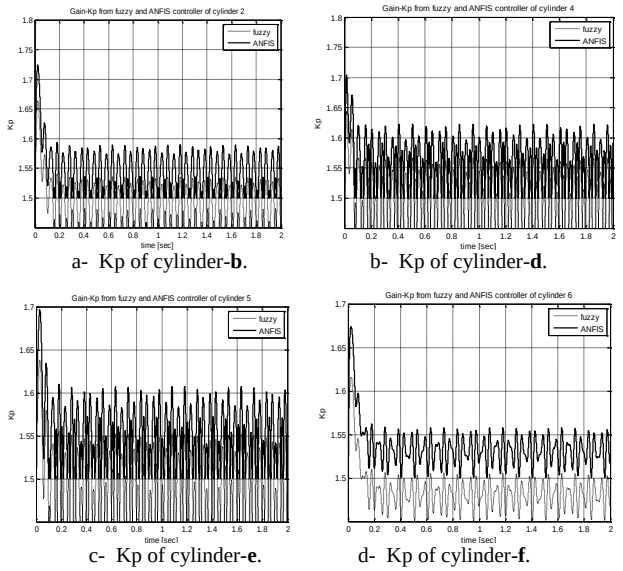
Three different controllers are implemented for computer simulation; the first one is the PID control. The second is the Fuzzy logic for tuning the gains of the PID controller (to find the optimal values) based on the expert's experience, where $K_{d,max}$=diag (3000, 3000, 3000, 3000, 3000, 3000), $K_{d,min}$=diag (1000, 1000, 1000, 1000, 1000, 1000), $K_{p,max}$=diag (2.5, 2.5, 2.5, 2.5, 2.5, 2.5), and $K_{p,min}$=diag (0.5, 0.5, 0.5, 0.5, 0.5, 0.5). In the Third, the parallel–implemented ANFIS technique is used for tuning the PID gains using the input output data from the fuzzy-PID as training data for each actuator for tracking the trajectory. During the simulations, the sampling period is chosen as 0.0005 s. Consequently, 50000 steps are included in every control simulation.



a- Initial MFs for each cylinder. b- Final MFs of cylinder-**c**.

c- Final MFs of cylinder-**e**. d- Final MFs of cylinder-**f**.

Figure 10. Initial and final (after training) membership functions MFs for input error.



a- Kp of cylinder-**b**. b- Kp of cylinder-**d**.

c- Kp of cylinder-**e**. d- Kp of cylinder-**f**.

Figure 11. Comparison between Kp gain for PID controller by fuzzy and ANFIS tuning methods.

Fig. 10 represents the initial and final (after training) membership functions for the error E for cylinders c, e, and f by the ANFIS method. From Figs 11, and 12, the PID gains (Kp and K_d) in both the cases of ANFIS and

Fuzzy tuning are not constant during the simulation, as in the case of only PID controller. The difference in the pressures between the two champers of each cylinder, which can be seen in Fig.13, are drawn to compare between PID controller method and the ANFIS PID tuning method, which indicates actual values for pressure differences for optimal trajectory tracking, where the effective piston area is 0.0015 m^2. Comparing Fig.14 with Fig.15, it can be observed that the end effector tracks the desired trajectory better with the ANFIS PID controller, since the control parameters Kp and K_d can be adjusted through the ANFIS network's learning. All the results demonstrate that the ANFIS PID control is better than Fuzzy PID and more effective than conventional PID controller.

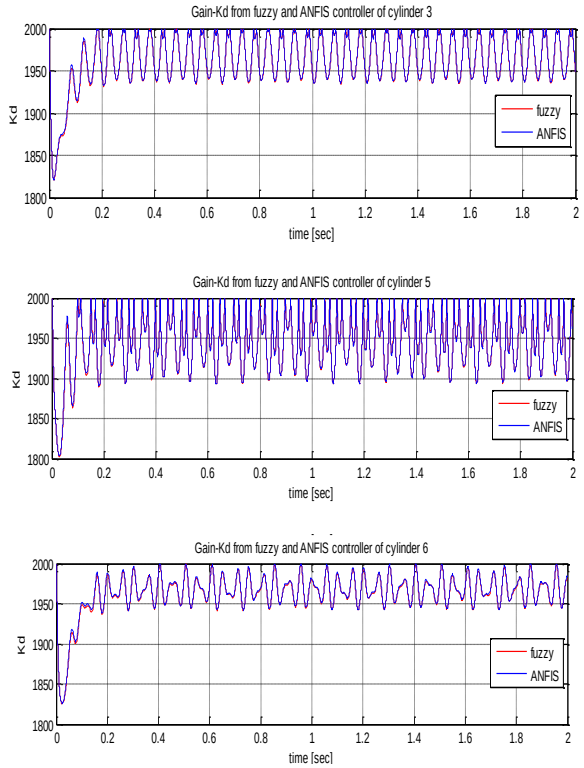
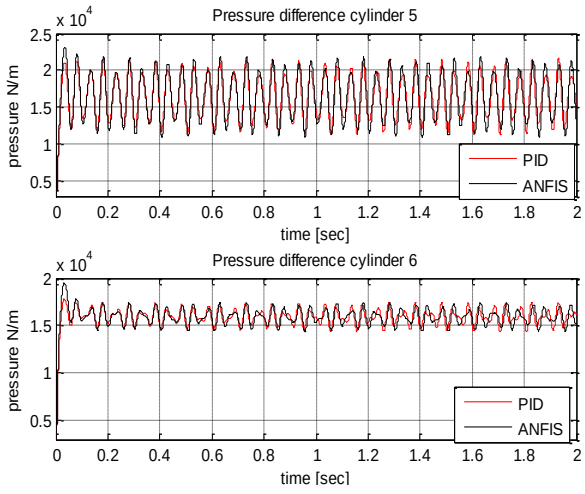


Figure 12. Comparison between the difference in the pressures resulted by conventional PID controller and ANFIS tuning method.



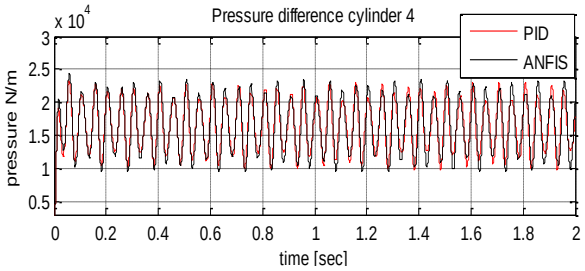


Figure 13. Comparison between the difference in the pressures resulted by conventional PID controller and ANFIS tuning method.

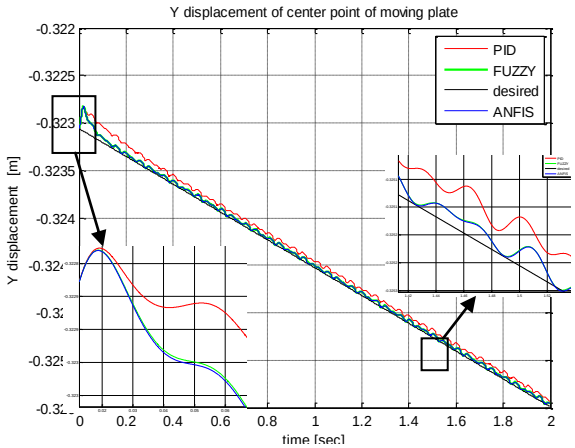


Figure 14. Y-displacement of moving plate (end effector).

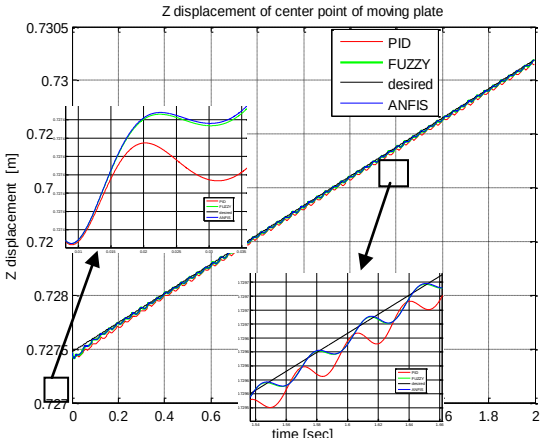


Figure 15. Z-displacement of moving plate(end effector).

VII. CONCLUSION

Parallel robots have higher rigidity and accuracy. In this paper, a 6-DOF hydraulically actuated parallel robot was investigated. Thus far, the PID controller has been used to operate under difficult conditions in this system, but since the gains of manual PID controller have to be tuned by trial and error procedures, obtaining optimal PID gains is very difficult without control design experience.

In order to improve the trajectory tracking performance, the fuzzy control and an ANFIS algorithm were proposed to adjust the parameters of the PID control. To evaluate the performance of the proposed control algorithms, they

were compared with the simple PID control. The two controllers were respectively used to control the end effector along a desired path. Simulation results have shown that the two methods for tuning the PID controller have better performance than the PID controller in terms of the reduction in position tracking errors of the end effector. Amongst the control schemes developed, ANFIS tuning has provided the best results for control of parallel robotic manipulators as compared to the conventional control strategies. The neuro-adaptive learning techniques provide a method for fuzzy modeling procedures to learn information about data sets. This technique makes the fuzzy logic capable of computing the membership function parameters that best allow the associated fuzzy inference system to track the given input and output data. ANFIS provides evident reductions in settling time, steady state errors. In conclusion, the ANFIS for tuning PID control represents a practical and valid alternative to parallel robots (control). This has been proved using MATLAB simulation of a parallel robotic manipulator.

Tuning method used in this system by ANFIS method has a good response without prior knowledge of the process. Also, by this method, more good responses than by the Fuzzy PID or only PID controllers are obtained. This control method is very useful to apply the process control system and helpful to select the most appropriate range for servovalves operation.

REFERENCES

[1] M. I. AL-Saedi, H. Wu, and H. Handroos, "Flexible Multibody Dynamics and Control of a Novel Hydraulically Driven Hybrid Redundant Robot Machine," in *Proc. IEEE 2nd International Conference on Applied Robotics for the Power Industry CARPI 2012*, Zurich, Switzerland, September 11-13, 2012, pp. 159-164.

[2] I. and A. Santoso, "Design and Control of the Stewart Platform Robot," in *Proc. the Third IEEE Asia International Conference on Modeling & Simulation*, 2009, pp. 475-480.

[3] B. R. Hopkins and R.L. Williams II., "Kinematics, design and control of the 6-PSU platform," in *Proc. Industrial Robot*, 2002, vol. 29, no. 5, pp. 443-451.

[4] Z. Yongsheng, LIU Zhifeng, C. Ligang, and Y. Wentong, "Enhanced Fuzzy Sliding Mode Controller for a 3-DOF Parallel Link Manipulator," in *Proc. 2nd International Asia Conference on Informatics in Control, Automation and Robotics CAR*, 2010, pp. 167-171.

[5] A. Akbas, "Application of Neural Networks to Modeling and Control of Parallel Manipulators," in *Parallel Manipulators, New Developments*, J. H. Ryu, Vienna, Austria: I-Tech Education and Publishing, April 2008, ch. 2, pp. 21-40.

[6] Y. Li, Y. Wang, and Z. Chen, "Research on Trajectory Tracking of a Parallel Robot Based on Neural Network PID Control," in *Proc. IEEE International Conference on Automation and Logistics*, Qingdao, China, Sept. 2008. pp. 504-508.

[7] L. Tian, "Intelligent Self-Tuning of PID Control for the Robotic Testing System for Human Musculoskeletal Joints Test," *Annals of Biomedical Engineering*, vol. 32, no. 6, pp. 899-909, June 2004.

[8] S. Ravi, M. Sudha, and P. A. Balakrishnan, "Design of Intelligent Self-Tuning GA ANFIS Temperature Controller for Plastic Extrusion System," *Journal of Modeling and Simulation in Engineering*, vol. 2011, Article ID 101437, 8 pages. 2011.

[9] A. A. Aldair, "FPG based ANFIC for full vehicle nonlinear active suspension systems," *IJAIA*, vol. 1, no. 4, pp. 1-15, October, 2010.

[10] D. Adhyaru, J. Patel, and R. Gianchandani, "Adaptive Neuro-Fuzzy Inference system based control of Robotic Manipulators," in *Proc. 2010 IEEE International Conference on Mechanical and Electrical Technology*, 2010, pp. 353-358.

[11] O. Bachir and A. F. Zoubir, "Adaptive Neuro-fuzzy Inference System Based Control of Puma 600 Robot Manipulator," *International Journal of Electrical and Computer Engineering*, vol. 2, no. 1, pp. 90-97, February 2012.

[12] T. Ngo, Y. N. Wang, T.L. Mai, M.H. Nguyen, and J. Chen, "Robust Adaptive Neural-Fuzzy Network Tracking Control for Robot Manipulator," *International J. Computer Commun. and Cont.*, vol. 7, no. 2, pp. 341-352, June, 2012.

[13] A. A. Shabana, *Dynamics of Multibody Systems*, third edition, Cambridge University Press, UK, 2005.

[14] K. M. Passino and S. Yurkovich, *Fuzzy Control*, Addison Wesley Longman, Inc., 2725 Sand Hill Road, Menlo Park, California 94025, USA, 1997.

[15] J. Jang, C. Sun, and E. Mizutani. *Neuro-Fuzzy and Soft Computing*, Prentice Hall, Upper Saddle River, NJ, USA, 1997.

Mazin I. Al-saedi is a doctoral student in Lappeenranta University of technology, Lappeenranta, Finland. He had M.Sc. degree in Applied Mechanics in 2005. His research interests include flexible multibody dynamics, robotics and control systems.

Huapeng Wu earned his B.Sc. in the field of fluid power transmission and control and M.Sc. in the field of flexible machine manufacturing system respectively in 1986 and 1993 from the School of Mechanical Engineering, Huazhong University of Science and Technology (HUST), China. He received the D.Sc. (Tech.) degree from LUT, with the topic "Design and Control of a Parallel Robot" in 2001. His specialties range from production machinery design to parallel robotics. He currently holds a robotics associate professorship position in the Laboratory of Intelligent Machines of LUT.

Heikki Handroos earned his M.Sc and D.Sc. (Tech.) degrees from Tampere University of Technology, Finland, 1985 and 1991. He has carried out research on mechatronics for the past 25 years ranging from the modeling of hydraulic systems to the control and development of large scale serial and parallel manipulators. He has published about 180 publications and led several academic, industrial, and EU-funded projects on mechatronics. He has been a member of the ASME Dynamic Systems and Control Division since 1990. He is full professor and Head of the Laboratory of Intelligent Machines, LUT.