Appl. Math. and Comp. Sci., 1996, vol.6, No.3, 595–616

FUZZY NEURAL HYBRID POSITION/FORCE CONTROL FOR ROBOT MANIPULATORS

KAZUO KIGUCHI*, TOSHIO FUKUDA**

Hybrid position/force control is one of the most important and fundamental control methods of robot manipulators. However, there are some problems in providing a hybrid position/force controller for practical use since conventional controllers are not able to adapt to an unknown environment. Recently, a lot of research has been carried out on fuzzy neural control, the combination of neural networks control and fuzzy control, in order to make the controllers intelligent. The fuzzy neural controller is expected to perform more sophisticated control than a conventional one in an unknown environment owing to its adaptation ability. In this paper, we propose a fuzzy neural hybrid position/force control for robot manipulators in an unknown environment using fuzzy logic, neural network, and fuzzy neural network. Simulations have been done with the use of a 3DOF planar robot manipulator to confirm the effectiveness of the proposed method.

1. Introduction

Recently, many researchers have applied both fuzzy logic and neural networks to control (Fukuda and Shibata, 1991; Kiguchi and Necsulescu, 1993; Mamdani, 1974; Popovic *et al.*, 1995; Shibata *et al.*, 1992; Yabuta and Yamada, 1990). It is known that fuzzy control (Mamdani, 1974; Popovic *et al.*, 1995) is capable of dealing with human knowledge. Therefore, the precise mathematical models of the plant and the environment are not required for designing the controller. There are some difficulties, however, to design the fuzzy controller systematically. Furthermore, once fuzzy control rules and membership functions are decided, usually they cannot be modified on-line even if the controller is not perfect. On the other hand, it is known that neural networks control (Fukuda and Shibata, 1991; Kiguchi and Necsulescu, 1993; Shibata *et al.*, 1992; Yabuta and Yamada, 1990) has an ability to learn from its experiments and adapt to a new environment on-line. Because of these abilities, this controller is especially effective in the case when the robot manipulators have to work in an unknown environment. It is difficult, however, to design a good neural network controller without learning. A robot manipulator might cause some damage to the environment before the controller adapts to the unknown environment. Furthermore, the meaning

* Niiagata College of Technology, 5–13–7 Kamishin'eicho, Niiagata 950–21, Japan, e-mail: kkiguchi@po.niigata-ct.ac.jp

** Dept. of Micro System Engineering, Nagoya University, 1 Furo-cho, Chikusa-ku, Nagoya 464-01, Japan, e-mail: fukuda@mein.nagoya-u.ac.jp

of each weight value of neural networks is not understandable for us. Therefore, using fuzzy neural control, a combination of fuzzy control and neural network control, seems to be one of the best ways to solve these problems (Kiguchi and Fukuda, 1995). Fuzzy neural networks have been applied in many fields of robotics (Kiguchi and Fukuda, 1995; Sarkodie-Gyan and Willumeit, 1995; Suh and Kim, 1994) in order to make the robots intelligent.

Robot manipulators are expected to control the force and position precisely. In order to control both the force and position simultaneously, a great deal of work has been carried out to elaborate the effective position/force control algorithms. One of the most important and fundamental control algorithms for robot manipulators is the hybrid position/force control (Raibert and Craig, 1981; Yoshikawa, 1987). The basic idea is separating the directions for the force control, which is normal to the constraint surface of the environment, and for the position control, which moves along the constraint surface in the Cartesian coordinate system. The hybrid position/force control, however, has not been applied in practice because of some difficulties. For example, the dynamics of the unknown environment affects the stability of the whole system while the robot manipulator is in contact with the environment. Furthermore, friction between the end-effector of the robot manipulator and the environment occurs since the robot manipulator is in contact with the environment. This problem does not exist if the robot manipulator controls the position only. The friction between the end-effector of the robot manipulator and the environment has to be compensated when the position of the end-effector is controlled along the surface of the environment as the force is applied simultaneously to the environment. Moreover, the friction force varies according to the applied force to the environment. Therefore, in the experiments of position/force control carried out by some researchers, something like a wheel is attached to the end-effector to be able to slide along the surface of the environment without friction (Lu and Goldberg, 1995). In this paper, we propose an intelligent hybrid position/force controller in order to solve these problems using fuzzy logic, neural networks, and fuzzy neural networks.

For the problem of the force control law of the hybrid control described above, a fuzzy neural force controller is applied in order to avoid unexpected overshooting at the beginning, compensate external disturbance, and make the controller adapt to the unknown environment using its adaptive ability. However, if the environment is much harder than the estimated environment, unexpected overshooting might happen. In this paper, a neural network is applied for adjusting the input information to the fuzzy neural force controller in order to adjust the controller to the environment instantly when the environment is much harder or softer than the estimated one. Furthermore, a fuzzy-controlled evaluation function is proposed for the effective learning algorithm of the fuzzy neural force controller.

For the problem of the position control law of the hybrid control described above, a fuzzy neural position controller is applied in order to compensate the external disturbance, modeling error, etc. In this paper, a specialized neuron for friction compensation, which is attached to the fuzzy neural network for the position control, is proposed to compensate the friction between the end-effector of the robot manipulator and the environment.

The effectiveness of the proposed method is verified by computer simulation with a 3DOF planar robot manipulator which is shown in Fig. 1.

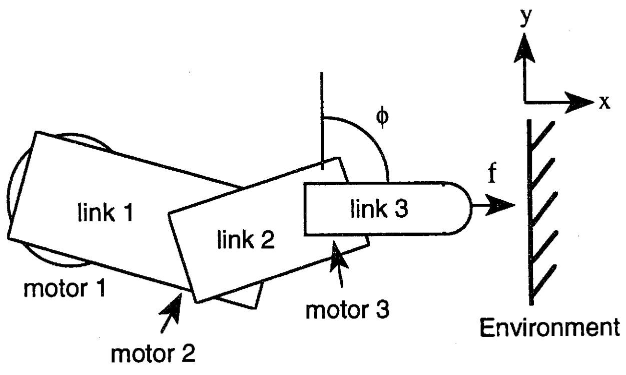


Fig. 1. 3DOF planar robot manipulator.

2. Hybrid Position/Force Control

In order to control the force applied to the environment with the robot manipulator shown in Fig. 1, and to control the position in the y-direction and the angle of the end-effector with respect to the environment simultaneously, a hybrid position/force controller seems to be appropriate (Raibert and Craig, 1981; Yoshikawa, 1987). The basic idea of the hybrid control is to separate the directions for the force control that is orthogonal to the constraint surface of the environment, and for the position control that moves along the constraint surface in the Cartesian coordinate system.

The dynamics equation of the planar robot is as follows:

$$M(q)\ddot{q} + h(q,\dot{q}) + F_{jc}\,\mathrm{sgn}(\dot{q}) = \tau - J^T f \tag{1}$$

where M is the inertia matrix, h denotes the Coriolis and centrifugal components, F_{jc} is the Coulomb friction of the robot manipulator joint, τ is the output torque, J stands for the Jacobian, f denotes the force applied to the environment, and q is an angular position vector.

From eqn. (1), we obtain the required motor torque equation

$$\tau = M(q)\ddot{q} + h(q,\dot{q}) + F_{jc}\,\mathrm{sgn}(\dot{q}) + J^T f \tag{2}$$

The acceleration of the end-effector of the robot manipulator in the Cartesian coordinate system is written as

$$\ddot{x} = J\ddot{q} + \dot{J}\dot{q} \tag{3}$$

where x is a position vector in the Cartesian coordinate system.

From eqns. (2) and (3) with the selection matrix S, which selects the directions for the force control that is normal to the constraint surface of the environment and for the position control that moves along the constraint surface in the Cartesian coordinate system, the following equation can be written for the hybrid control:

$$\begin{aligned}\tau &= M(q)J^{-1}\left[(I-S)u_p - \dot{J}\dot{q}\right] + h(q,\dot{q}) + F_{jc}\,\mathrm{sgn}(\dot{q}) \\ &\quad + J^T f + J^T S u_f \end{aligned} \qquad (4)$$

where u_p is a command vector for position control and u_f is a command vector for force control. In this paper, $u_p = u(x_d, x, f)$ is generated by the fuzzy neural position controller in order to reduce the error between the desired position (trajectory) and the measured position (trajectory), where x_d is the desired manipulator position, x is the measured manipulator position, and f is the measured force. Here $u_f = u(f_d, f, M_0, x)$ is generated by the fuzzy neural force controller in order to reduce the error between the desired force and the measured force, where f_d is the desired force, f is the measured force, M_0 is the manipulator momentum in the direction of the force control in the Cartesian coordinate, and x is the manipulator position.

The block diagram of the hybrid controller is shown in Fig. 2. This controller controls the force applied to the environment, the position in the y-direction, and the angle of the end-effector (see Fig. 1).

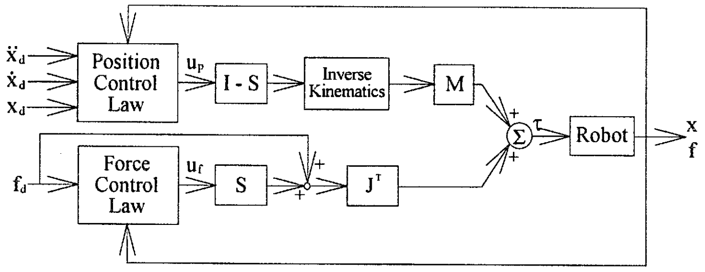


Fig. 2. Block diagram of hybrid control.

In this paper, the resolved acceleration control method (Luh *et al.*, 1980), in which the acceleration is specified and all the feedback control is carried out in the Cartesian coordinate system, is applied for the control of the angle of the end-effector since any adaptation is not required for the end-effector angle control. The force command for the end-effector angle in the Cartesian coordinate system u_{pa}, one of the components of the command vector for position control u_p in eqn. (4), is written as

$$u_{pa} = \ddot{x}_{da} + K_v(\dot{x}_{da} - \dot{x}_a) + K_p(x_{da} - x_a) \qquad (5)$$

where x_{da} is the desired angle of the end-effector, x_a is the measured angle of the end-effector (a component of the position vector x). Moreover, K_v and K_p are the velocity and position gain matrices, respectively.

Obtaining the desired position/force with the hybrid controller using conventional control methods is usually difficult because of the following reasons. First, it is very difficult to make perfect mathematical models of the robot manipulator and the environment. Second, a high feedback gain for the force control should not be used, since the dynamics of the environment affects the stability of the whole system while the robot manipulator is in contact with the environment. Third, friction in the robot manipulator joints and friction between the robot manipulator and the environment cannot be compensated completely. Especially, in the case when the force applied to the environment varies, the amount of the Coulomb friction also changes over time. The equation of the Coulomb friction between the robot manipulator and the environment F_{er}, which acts in the direction of the position control in the Cartesian coordinates, is written as follows:

$$F_{er} = \mu_k f \tag{6}$$

where μ_k is the coefficient of kinematic friction, f is the force applied perpendicularly to the environment. Since it is difficult to obtain μ_k when a property of the environment is unknown, it is difficult to compensate the Coulomb friction between the robot manipulator and the environment with conventional control methods. Fourth, external disturbance cannot be avoided. Furthermore, if the environment is made of rubber, plastics, wood, etc., the property changes over time since it is affected by the temperature and humidity of the air. Consequently, adaptation abilities are required for the hybrid controller to overcome these problems. In this paper, adaptation abilities are realized in the hybrid controller by applying fuzzy logic, a neural network, and fuzzy neural networks in order to generate the adaptive control output u_p and u_f in eqn. (4) to solve these problems.

Practically, most modelling errors or uncertainties are included in the definition of the amount of mass and friction in eqn. (4). These modelling errors or uncertainties, and unmodelled external disturbances are compensated by adjusting u_p and u_f.

The details of the proposed controller are explained in Sections 3 and 4.

3. Force Control Law of the Hybrid Control

In order to apply the desired force to an environment whose dynamic property is unknown, the adaptive control, such as the adaptive-type neural network control, has to be applied. The adaptive-type neural network controller, however, cannot adapt to the unknown object immediately (Kiguchi and Necsulescu, 1993). Consequently, overshooting which gives too much force to the object might happen. The authors have proposed the fuzzy neural force controller (Kiguchi and Fukuda, 1995) for robot manipulators to solve this problem. In order to make the fuzzy neural force controller, the fuzzy force controller which is able to avoid unexpected overshooting has to be designed first using our knowledge. Then it is converted to a neural network which is able to adapt to an unknown environment using the back-propagation learning algorithm. However, if the environment is much harder than the estimated environment, overshooting cannot be avoided even with the fuzzy neural network force control. In this paper, we propose to use a neural network which learns off-line to adjust the input information to the fuzzy neural network force controller. By adjusting the input information, the controller becomes capable of dealing instantly with a harder or softer environment than estimated. We call this neural network the Input Adjustment Neural Network (IANN). The architecture of the fuzzy neural force controller is depicted in Fig. 3. Here Σ means the sum of the inputs, and Π means the multiplication of the inputs. The fuzzifier layer, in which each neuron represents a membership function for the input, consists of 10 neurons, the rule layer consists of 17 neurons, and the defuzzifier layer consists of 2 neurons.

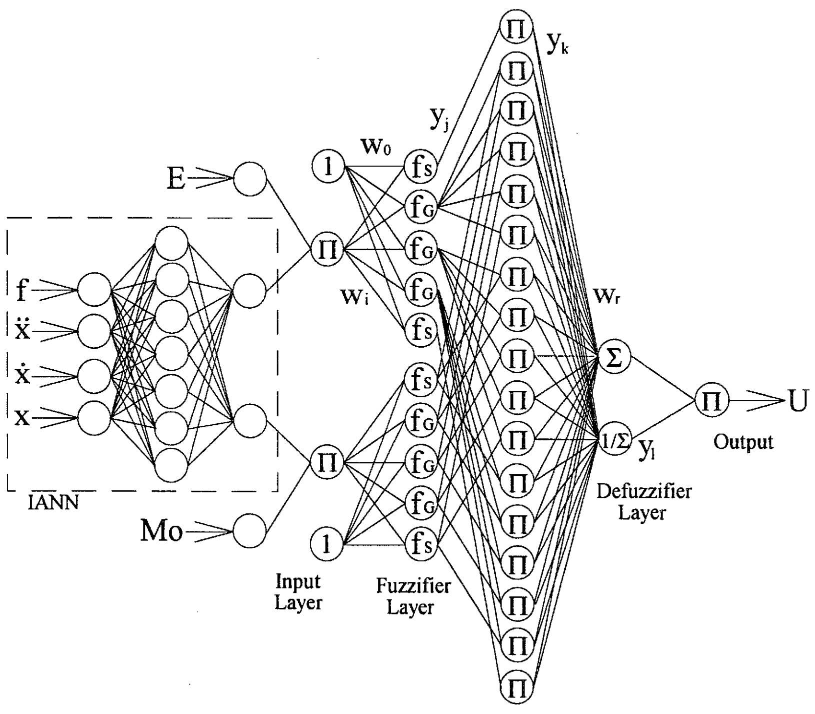


Fig. 3. Architecture of the fuzzy neural force controller.

3.1. Fuzzy Neural Force Control

Usually fuzzy control uses the error and its change rate for the input to the controller. In the case of force control, it is not easy to use the error change rate since the signals from a force sensor are noisy. In this case, the manipulator's pushing velocity against the environment might be used instead of the force error change rate. As far as robot manipulators are concerned, however, the sense of the velocity depends on the amount of their inertia. This means that the sense of velocity might be different even with the same robot manipulator if its configuration is different. Therefore, the same membership functions of the velocity should not be used if the configuration of the robot manipulator is different. In order to avoid this kind of problems, the authors proposed to use the momentum of the robot manipulator instead of the velocity of the robot manipulator (Kiguchi and Fukuda, 1995). It is also a good method to use the kinetic energy of the robot manipulator instead of the velocity. However, dealing with the squared velocity for the input to the controller is difficult. Therefore the momentum of the robot manipulator, whose equation is written down below, is used in this paper. Namely, we have

$$M_0 = M_x(q)v \tag{7}$$

where M_0 represents the momentum of the manipulator in the direction of the force control, v denotes the velocity of the robot manipulator in the direction of the force control, $M_x(q)$ stands for the inertia matrix in the Cartesian coordinates when the angular position vector of the robot manipulator is q. The inertia matrix in the Cartesian coordinates $M_x(q)$ is written as the following equation with the use of the inertia matrix in the manipulator angular coordinates $M(q)$ and the Jacobian J:

$$M_x(q) = J^{-T}(q)M(q)J^{-1}(q) \tag{8}$$

There are five kinds of fuzzy numbers (PB, PS, ZO, NS, and NB) for each input variable (the error and manipulator momentum). The membership functions are obtained through the fuzzifier layer of the neural network using the Gaussian function, which is written down as eqn. (9), and the sigmoidal function, which is written down as eqn. (11). The simple fuzzifier layer is realized by using these two kinds of functions:

$$f_s(u_s) = \frac{1}{1+e^{-u_s}} \tag{9}$$

$$u_s(x) = w_0 + w_i x \tag{10}$$

$$f_G(u_G) = e^{-u_G^2} \tag{11}$$

$$u_G(x) = \frac{w_0 + x}{w_i} \tag{12}$$

where w_0 is a threshold value and w_i is a weight. In the Gaussian function, w_0 is the mean value and w_i is a deviation of the membership function.

The calculated membership functions from the fuzzifier layer are sent to the rule layer and multiplied in the neuron in the rule layer according to fuzzy IF-THEN rules. There are two outputs from each neuron in the rule layer. One of them is multiplied by the weight and summed up in the next layer. The other is just summed and then inverted in the defuzzifier layer. The multiplied values of these outputs will be the output of the fuzzy-neural network. Note that the output of the controller is a real value, not fuzzy. The output from the controller is the force command in the Cartesian coordinates.

3.2. Input Adjustment Neural Network (IANN)

It is possible through the fuzzy neural force control method described in Section 3.1, to apply the desired force trajectory to almost every environment without unexpected overshooting, assuming the property of the environment is hard when we design the fuzzy force control law. However, if the environment is much harder than the estimated one, unexpected overshooting might happen even with the fuzzy neural force control. If it is possible to estimate the property of the environment in the early stage of the force control from the relation between the deformation of the environment and the reaction force. The definition of the membership function of the fuzzy neural force controller can be adjusted immediately according to the property of the environment in order to avoid unexpected overshooting. In this paper, we propose the Input Adjustment Neural Network (IANN) for the purpose of adjusting two input-information to the controller, the error between the desired and measured force E and the manipulator momentum M_0, from the relation between the deformation of the environment and the reaction force in order to make the controller adapt immediately to the environment.

Adjusting the membership function of the input variables, which is defined when the fuzzy force control law is designed, means multiplying the membership function of the input variables by adjustment coefficients (the output from the IANN). It produces the same effect as making the shape of the membership function wider or narrower (see Fig. 4). This means that the IANN is able to change the definition of the membership function immediately. In other words, the antecedent of the fuzzy force control law is adjusted immediately according to the property of the environment by the IANN. For example, if the environment is harder than estimated, the error between the desired force and measured force is converted by the IANN to be smaller, and the manipulator momentum is converted to be larger before they are used as input information of the controller. This operation makes the controller control precisely if the error is close to zero even though the actual error is still large, and respond to the larger manipulator momentum than the actual amount.

For input variables to the IANN, the deformed (pushed) distance, deforming velocity, deforming acceleration of the environment, and reaction force from the environment are used since the environment can be modelled as the following equation assuming it is a mass-spring damper system:

$$f = M_e \ddot{x}_e + B_e \dot{x}_e + K_e x_e \tag{13}$$

where f is the reaction force from the environment, and x_e is the surface displacement of the environment. The outputs from the IANN are the adjustment coefficient K_E for the error information and the adjustment coefficient K_{M_0} for the manipulator momentum. These adjustment coefficients of the input variables to the controller vary between 0 and 2. If the property of the environment is the same as that of the estimated one, the adjustment coefficients become 1.

The IANN learns off-line using the back-propagation learning algorithm. Several properties of the information regarding the environment, which represent the variety of materials, are prepared for the training data of the IANN. The IANN is trained to make K_E smaller than 1 and K_{M0} larger than 1 in accordance with the environment property, if the environment is harder than the estimated one. If the environment is softer than the estimated one, the IANN is trained to make K_E larger than 1 and K_{M0} smaller than 1 in accordance with the environment property.

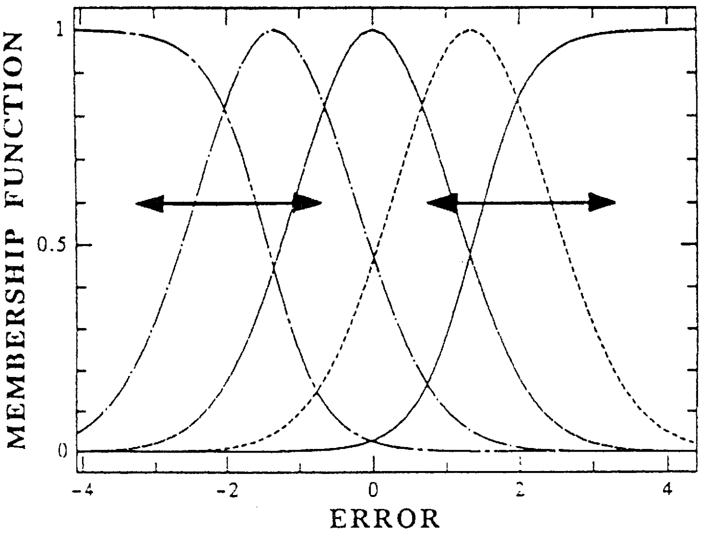


Fig. 4. Change of shape of membership functions.

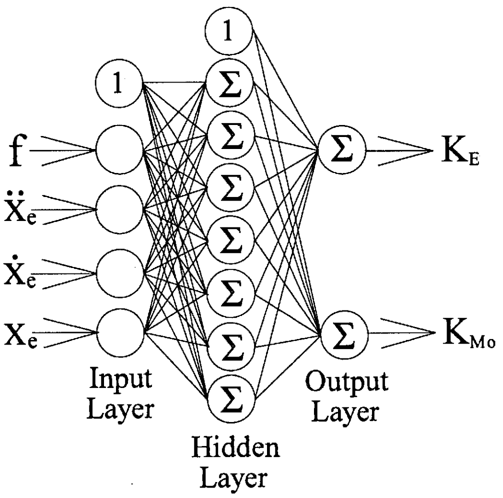


Fig. 5. Architecture of the Input Adjustment Neural Network (IANN).

3.3. Learning Algorithm of the Force Controller

Every weight of the fuzzy neural force controller is adjusted at every sampling time moment by the back-propagation learning algorithm to minimize the evaluation function. The squared error is usually used for the evaluation function:

$$Y(k) = \frac{1}{2}\left[f_d(k) - f(k)\right]^2 \tag{14}$$

where f_d represents the desired force, and f the measured force.

When the desired force is the combination of periodic step signals, however, a little overshooting might occur after some period of time if only the force error is evaluated for learning of the fuzzy neural force controller. It takes a few milliseconds to catch up with the suddenly changed desired force since no desired force trajectory exists, while the controller keeps learning to output a larger force command to the robot manipulator in order to catch up with the desired force as quickly as possible. In this paper, a fuzzy controlled evaluation function is introduced in order to avoid this problem. The proposed evaluation function is written as

$$\begin{aligned} Y(k) \;=\; & \frac{1}{2}\Big\{\left[f_d(k) - f(k)\right]^2 \\ & + K_z\left[(f_d(k) - f(k)) - (f_d(k-1) - f(k-1))\right]^2\Big\} \end{aligned} \tag{15}$$

where K_z is the fuzzy controlled variable of a number which is large enough to make the velocity term dominate the evaluation function when the error is zero and become 0 when the error is far from zero. This means the expression inside the second brackets in eqn. (15), which is the squared error difference between the error at present and previous time step, is evaluated only when the error is almost zero. Therefore this part works to prevent an overshooting as the error approaches zero, since it tries to constrain the robot manipulator movement when the desired force does not change. The input variables of this evaluation function fuzzy controller are the same as those of the fuzzy neural force controller. The output of this fuzzy controller is K_z.

The equation of individual weight adjustment in the defuzzifier layer is

$$\Delta w_r = \eta(f_d - f)y_l y_k \tag{16}$$

where η is the learning rate of the fuzzy neural controller.

The equation of individual weight adjustment in the fuzzifier layer is

$$\Delta w_i = \eta \sum_j \frac{\partial e_{ij}}{\partial y_{ij}} f'(u)u' \tag{17}$$

where y_{ij} is the output from the fuzzifier layer, e_{ij} is the output error in the fuzzifier layer, $f'(u)$ and u' are the derivatives of the activation function and input to the function (see eqns. (9)–(12)), respectively. Different learning rates are used for two input variables and rules, since the adjustment rates of the weights are different.

Even if an initially designed fuzzy force control law is not perfect, the compensation of the friction of the robot manipulator joints is not perfect, an external disturbance exists, or a property of the environment changes, the proposed force controller is capable of applying the desired force to an unknown environment with this learning algorithm.

4. Position Control Law of the Hybrid Control

In order to control the position of the robot manipulator which applies simultaneously the force to the environment, the Coulomb friction between the robot manipulator and the environment has to be compensated. The Coulomb friction varies according to the amount of the force applied to the environment. If the value of the friction coefficient is known previously, it might not be very difficult to compensate the friction even though it changes. However, it is difficult to obtain the coefficient of the friction between the robot manipulator and the unknown environment. The equation of the Coulomb friction F_{er} is written down as eqn. (6).

In this section, the details of the proposed fuzzy neural position control, which compensates the friction between the robot manipulator and the environment, are described. The architecture of the proposed fuzzy neural position controller is shown in Fig. 6. Here Σ denotes the sum of the inputs and Π stands for multiplication of the inputs. The fuzzy neural position controller consists of a fuzzy neural network division for trajectory control and a specialized neuron division for friction compensation. Here, the friction means the one between the robot manipulator and the environment. In this controller, these two divisions do not learn (adjust each weight value) simultaneously. In other words, learning is switched between the fuzzy neural network division for trajectory control and the specialized neuron division for friction compensation depending on the situation. The output from the proposed fuzzy neural position controller, the combination of the output from the fuzzy neural network division for trajectory control and from the specialized neuron division for friction compensation, is the force command for the robot manipulator in the Cartesian coordinate system.

4.1. Fuzzy Neural Position Control

The trajectory (movement) of the robot manipulator is controlled by a fuzzy neural network division for trajectory control (see Fig. 6) once it begins to move. Since the effect of the Coulomb friction is supposed to be canceled out by the specialized neuron division for friction compensation, this fuzzy neural network is in charge of the control of the robot manipulator trajectory only.

The first step to construct the fuzzy neural network for trajectory control is to design a fuzzy position controller using the knowledge of experts. The next step is to convert the designed fuzzy position controller to a neural network. Gaussian and sigmoidal functions are used as membership functions in the fuzzifier layer, similarly as in the fuzzy neural force controller. The fuzzy neural network for trajectory control consists of 10 neurons in the fuzzifier layer, 17 neurons in the rule layer, and 2 neurons in the defuzzifier layer.

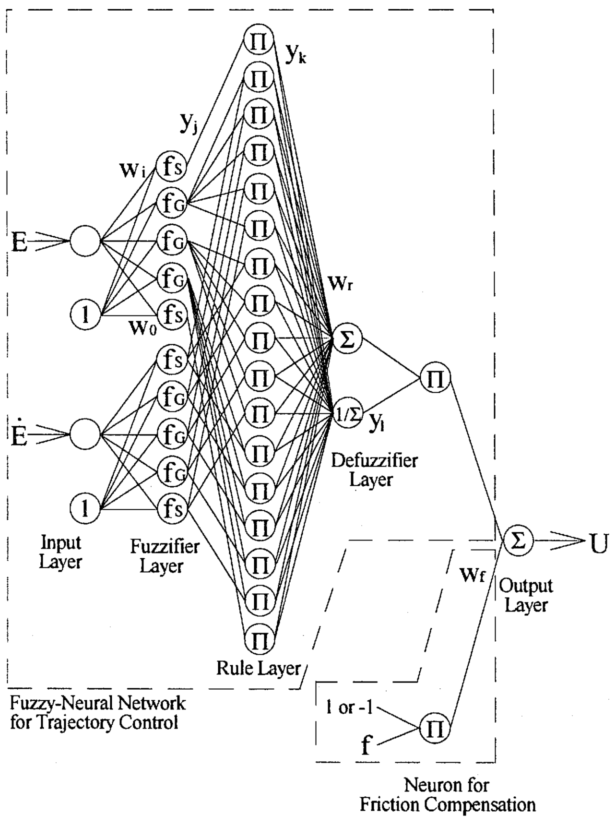


Fig. 6. Architecture of the fuzzy neurall position controller.

The principle of this fuzzy neural network is the same as that of the fuzzy neural force controller. Therefore we omit to explain how this fuzzy neural network works since we have already done this for the fuzzy neural force controller in the previous chapter. The outputs from the fuzzy neural network division for trajectory control and from the friction neuron division are summed up to make the force command for the robot manipulator in the Cartesian coordinate system.

4.2. Friction Neuron

In this subsection, we discuss a specialized neuron for friction compensation (we call this neuron a friction neuron) in an unknown environment. The force applied to the environment f and the direction (1 or -1) in which the friction compensation is needed are used as the input information to the friction neuron (see Fig. 6). They are multiplied in the friction neuron and then multiplied by the weight w_f. This becomes the output from the friction neuron. This output is the force command

5. Simulation

Computer simulations have been carried out in order to confirm the effectiveness of the proposed fuzzy neural hybrid position/force controller for an unknown environment with a 3DOF planar robot manipulator (Fig. 1). The angle of the robot manipulator's end-effector has been controlled to be always perpendicular to the environment with the use of the resolved acceleration control method. In the simulation, the sampling time has been set to 1 m per second.

5.1. Simulation for the Force Controller

In order to evaluate the proposed force control method, some simulations are performed against several properties of the environment. The environment model expressed by eqn. (13) is used for the simulation. The modelling error is included, while assuming a practical modelling error and unexpected external disturbance. The desired contact force applied to the environment is a combination of step signals changed between 20 N and 10 N every 2 s. First of all, some simulations were carried out with and without the IANN in order to see its effect. The simulation results against the environment model whose property is the same as that of the estimated one are shown in Fig. 8. One can see that the desired force can be obtained without unexpected overshooting with fuzzy control rules, and the error is reduced over time by the learning and adaptation abilities of the neural network in the proposed force controller. If the controller does not have learning and adaptation abilities, the error caused by the errors of the fuzzy control rules and the unexpected external disturbance will not be eliminated. In this simulation, the effect of the IANN is not seen in the results, since the property of the environment is the same as that of the estimated one.

Next, we changed all the coefficients of the environment model defined by eqn. (13) to be ten times smaller than the estimated property of the environment in order to verify whether the proposed controller is effective in a soft environment. The corresponding results are shown in Fig. 9. They show that the desired force can be realized with the proposed controller, even though the environment is much softer than the estimated environment although the IANN affects little the results for the soft environment.

For the next simulation, all the coefficients in eqn. (13) are changed to be ten times larger than the estimated property of the environment in order to verify whether the proposed controller is effective in a hard environment. The simulation results without the effect of the IANN are shown in Fig. 10. One can observe a large overshooting and oscillations. However, they can be avoided if the IANN is applied to the controller as shown in Fig. 11. This result shows that the IANN affects a lot the controller by adjusting the input information immediately in the case where the environment is harder than the estimated one.

In order to illustrate the effectiveness of the proposed fuzzy controlled evaluation function in learning of the force controller, another simulation was performed with and without the proposed evaluation function. If the controller learns for a long time to minimize the evaluation function of eqn. (13), a small overshooting might happen

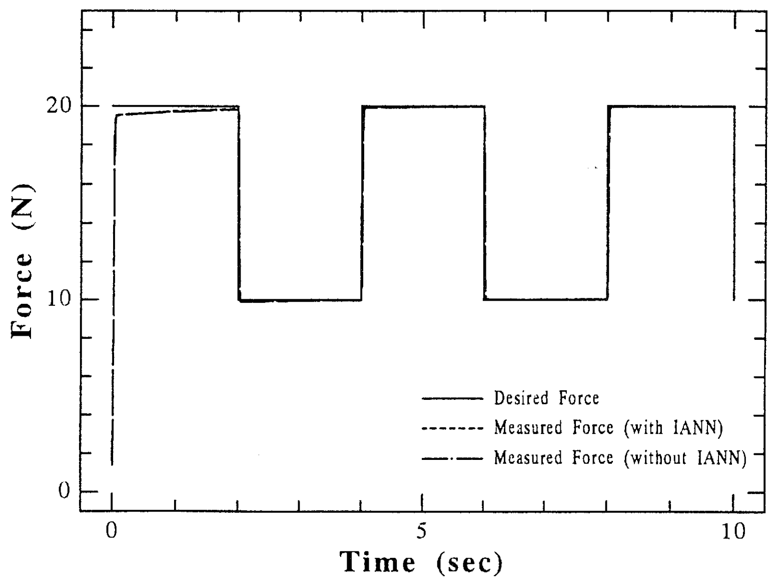


Fig. 8. Results of the force control simulation.

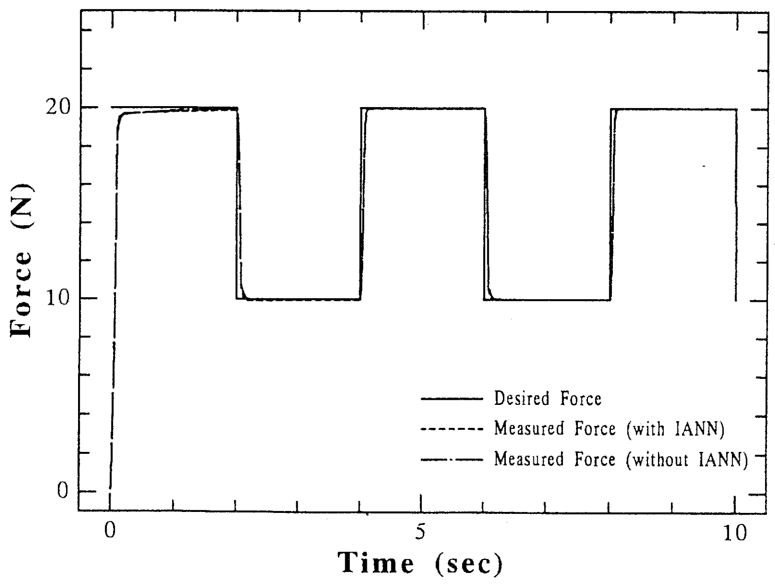
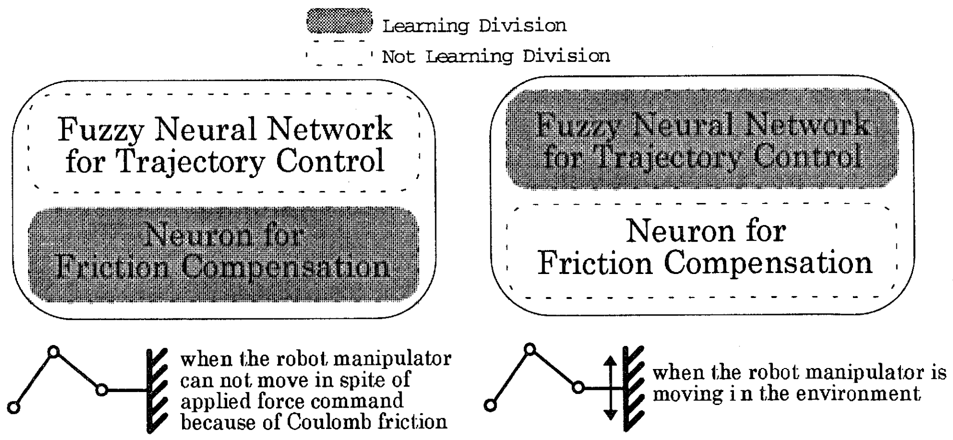


Fig. 9. Results of the force control simulation for a soft environment.

used for friction compensation. The friction neuron keeps learning until the robot manipulator begins to move. In other words, the weight of the friction neuron keeps increasing until the robot manipulator begins to move. Therefore the weight value will be almost the same as the value of the friction coefficient between the robot manipulator and the environment after some adaptation cycles. The output from this neuron becomes the output from the controller by adding the output from the fuzzy neural network for trajectory control which is explained in the previous section.

4.3. Learning Algorithm of the Position Controller

In order to adjust the value of each weight in the fuzzy neural network for trajectory control and the friction neuron, the back-propagation algorithm is adopted. Each weight is adjusted at every sampling time. We introduce a switch-learning algorithm in this section. The basic idea is switching the learning division between the fuzzy neural network division for trajectory control and the friction neuron division. The friction neuron division is able to learn only when the robot manipulator cannot move in the environment in spite of the driving force applied. Once the robot manipulator begins to move in the environment, the friction neuron division stops learning and the fuzzy neural network division for trajectory control begins to learn in turn. On the contrary, if the robot manipulator gets stuck in the environment though the controller is still applying the driving force, the fuzzy neural network division for trajectory control stops learning and the friction neuron division begins to learn again. This means that the weights of the fuzzy neural network division for trajectory control are not adjusted when the weight of the friction neuron division is adjusted, and the weight of the friction neuron division is not adjusted when the weights of the fuzzy neural network division for trajectory control are adjusted. This algorithm is shown in Fig. 7. While the robot manipulator is moving, the friction neuron keeps outputting



Learning Division
Not Learning Division
Fuzzy Neural Network for Trajectory Control
Neuron for Friction Compensation
Fuzzy Neural Network for Trajectory Control
Neuron for Friction Compensation
when the robot manipulator can not move in spite of applied force command because of Coulomb friction
when the robot manipulator is moving in the environment

Fig. 7. Switch-learning algorithm.

the force for friction compensation according to the value of the friction coefficient that has been learned. If the direction of the robot manipulator movement is changed, the direction of the force for friction compensation is also changed since the values $+1/-1$ of the input to the friction neuron change according to the direction of the robot manipulator movement.

The squared error between the desired and measured (calculated) positions is adopted for the output error function. The objective of the backpropagation algorithm is to minimize this function whose equation is

$$Y = \frac{1}{2}(x_d - x)^2 \tag{18}$$

where x_d is the desired position and x is the measured (calculated) position.

The equation of weight adjustment of the friction neuron in the case, where the direction in which the friction compensation is needed when it learns is 1, takes the form

$$\Delta w_f = \eta(x_d - x)f \tag{19}$$

where η is the learning rate of the friction neuron. If the direction is -1, the equation is changed to

$$\Delta w_f = -\eta(x_d - x)f \tag{20}$$

When the fuzzy neural network for trajectory control learns, the equation of individual weight adjustment in the defuzzifier layer of the fuzzy neural network for trajectory control is

$$\Delta w_r = \eta(x_d - x)y_1 y_k \tag{21}$$

where η is the learning rate of the fuzzy neural network.

The equation of individual weight adjustment in the fuzzifier layer is

$$\Delta w_i = \eta \sum_j \frac{\partial e_{ij}}{\partial y_{ij}} f'(u)u' \tag{22}$$

where y_{ij} is the output from the fuzzifier layer, e_{ij} is the output error in the fuzzifier layer, $f'(u)$ and u' are the derivatives of the activation function and input to the function, respectively.

Different learning rates are used for two input variables, rules of the fuzzy neural network division for trajectory control, and the weights of the friction neuron division, since the adjustment rates of the weights are different. Therefore four different learning rates are used in the controller.

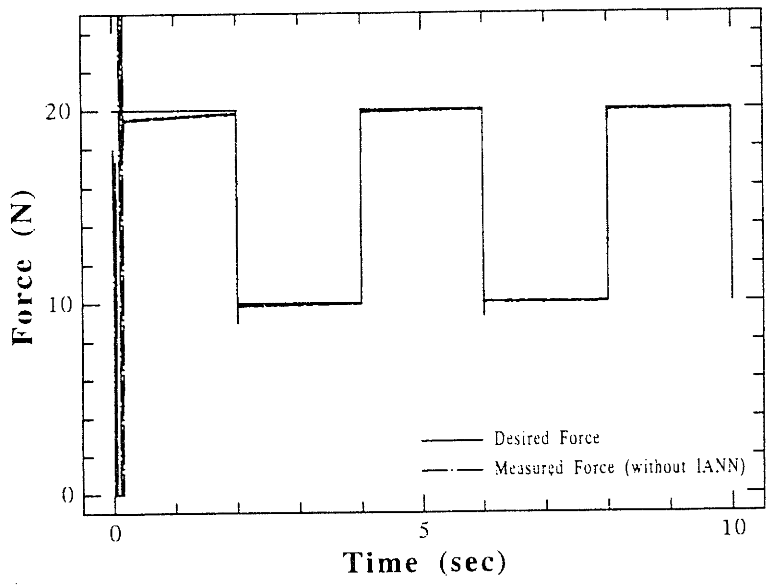


Fig. 10. Results of the force control simulation for a hard environment without IANN.

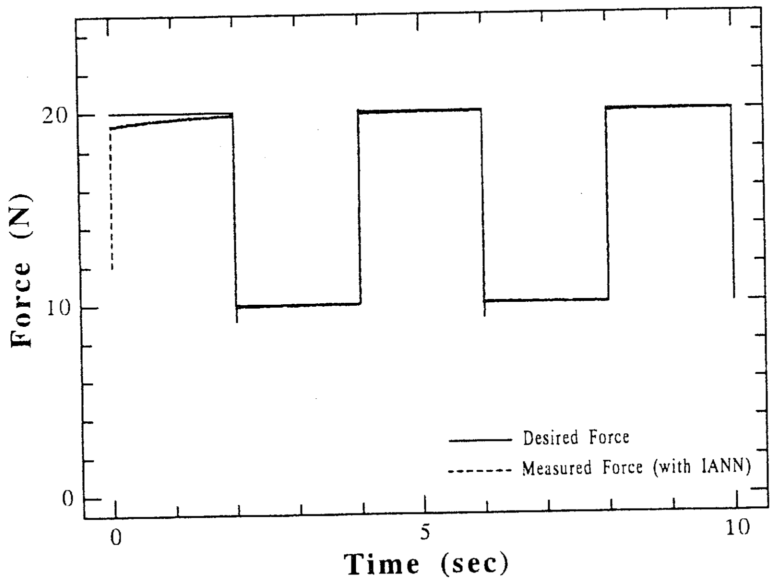


Fig. 11. Results of the force control simulation for a hard environment with IANN.

as shown in Fig. 12. In order to recognize this small overshooting, a magnified range of reaction force between 19.8 N and 20.2 N is used in Fig. 12. One can see that a small overshooting occurred after the controller has been learned for a long time (Fig. 12). This problem can be avoided using the proposed fuzzy controlled evaluation function in eqn. (15). The corresponding simulation results are shown in Fig. 13. They show that the overshooting can be avoided by using the proposed fuzzy controlled evaluation function.

These simulation results show that the desired force can be realized during the position/force control using the proposed force control method.

5.2. Simulation for the Position Controller

The robot manipulator applies the force to an unknown flat environment by pushing perpendicularly with the end-effector. The position of the end-effector of the robot manipulator is controlled along the environment surface horizontally (the y-direction in Fig. 1). The force and position are controlled simultaneously. There are four inputs to the proposed controller, namely the error between the desired and measured (calculated) positions, its changing rate, the direction (1 or -1) in which the friction compensation is needed, and the amount of the force applied to the environment. The output from the controller is the force command to the robot manipulator in the Cartesian coordinate system. The weights which belong to the learning division of the controller are adjusted at every sampling time.

The desired contact force applied to the environment during the position/force control is a combination of step signals changed between 20 N and 10 N every 2 s as shown in Fig. 8. The desired trajectory (position) in the y-direction along the surface of the environment during the hybrid position/force control is chosen as $0.05\sin(0.5\pi t)$ m.

In order to see the effect of the friction neuron, some simulations for trajectory control were performed with and without it. The simulation without the friction neuron means that the force command for the robot manipulator motion in the Cartesian coordinate system depends on the output from the fuzzy neural network division for trajectory control of the proposed controller only. In this case, the fuzzy neural network division for trajectory control keeps learning from the beginning of the control without switch-learning. The simulation results of the measured (calculated) trajectory which follows the desired trajectory during the position/force control with the 3DOF robot manipulator, in the case where the coefficient of kinematic friction is equal to 2.0, are shown in Fig. 14. One can observe that the measured trajectory with the effect of the friction neuron begins to follow the desired trajectory accurately in a short time. On the other hand, the measured trajectory without the effect of the friction neuron cannot follow the desired trajectory accurately even in the third period (one period is 4 seconds).

Another simulation has been performed to show that the proposed method is applicable with any amount of friction force. In this simulation, the coefficient of kinematic friction is increased to 4.0. The corresponding results are shown in Fig. 15.

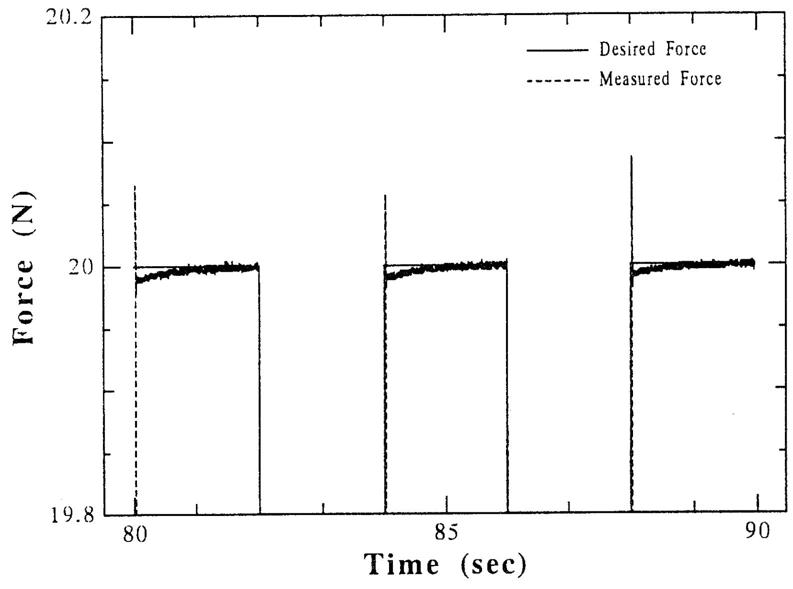


Fig. 12. Results of the magnified force control simulation.

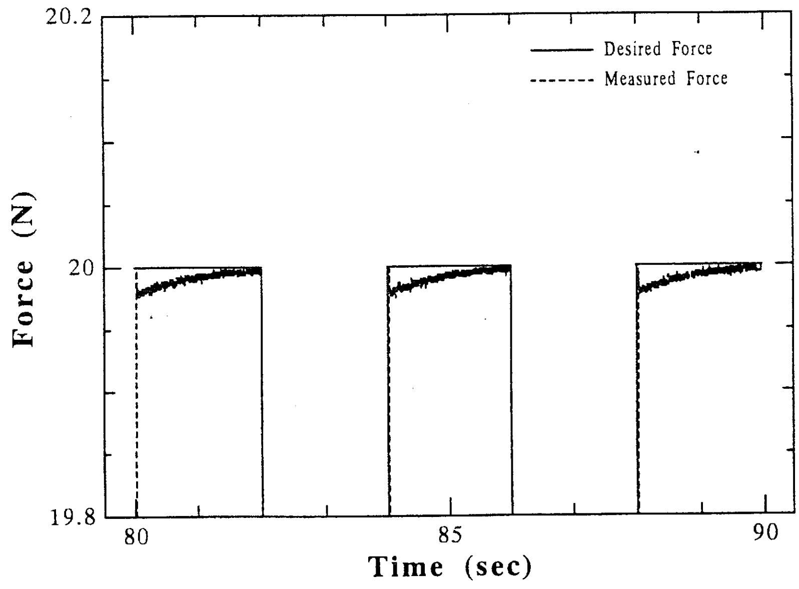


Fig. 13. Results of the modified force control minimizing the proposed evaluation function.

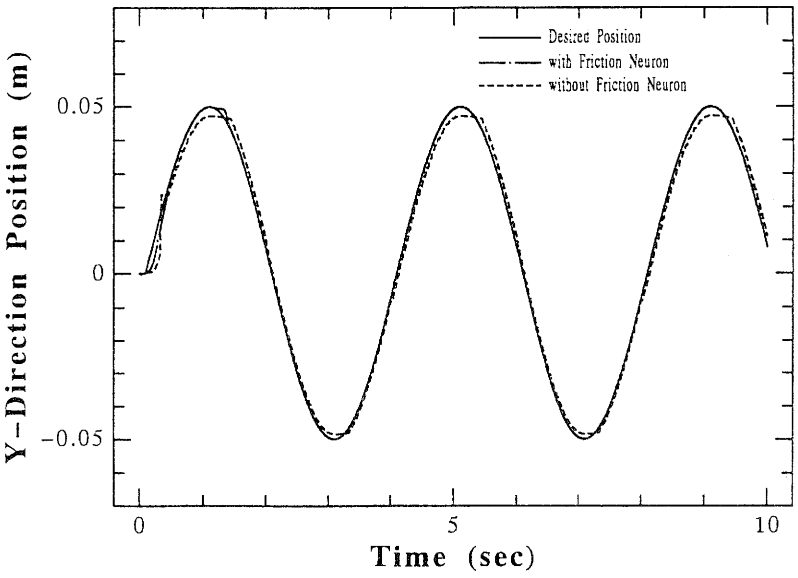


Fig. 14. Results of the position control simulation with the coefficient of kinematic friction equal to 2.0.

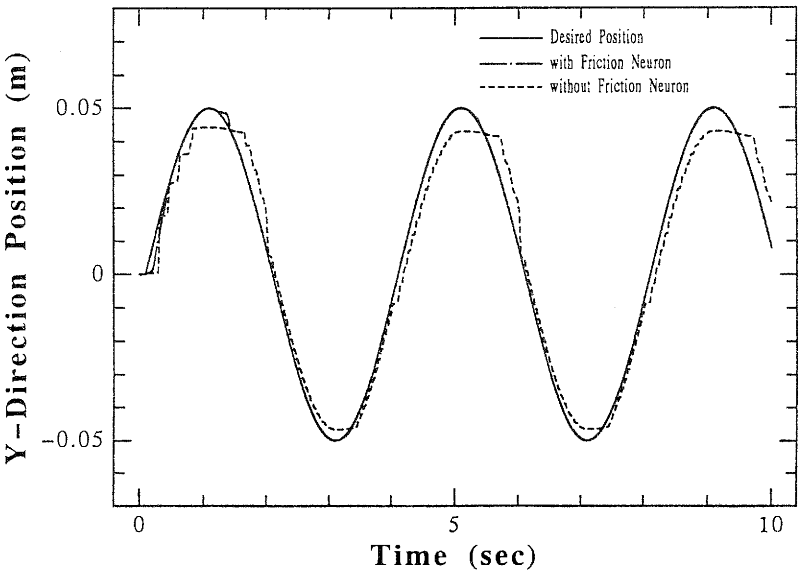


Fig. 15. Results of the position control simulation with the coefficient of kinematic friction equal to 4.0.

They reveal that the change in the coefficient of kinematic friction does not affect the proposed position controller very much. The results without friction neuron, however, become much worse than the previous ones, since the friction force is larger than before.

We can also see that the results without the friction neuron are affected by the change of the force applied to the environment ranging from 10 N to 20 N every 2 s. The results of the proposed position controller are not affected by the change of the applied force, since the friction force is compensated by learning the coefficient of kinematic friction.

6. Conclusions

A hybrid position/force control method for an unknown environment, which makes use of neural networks, fuzzy logic, and fuzzy neural networks, is introduced in the paper. The Input Adjustment Neural Network is proposed to adjust the input information to the fuzzy neural force controller according to the property of the environment in order to make the controller adapt to the environment immediately. Furthermore, an effective learning algorithm is proposed which uses a fuzzy controlled evaluation function.

In order to control the position of the robot manipulator which applies simultaneously the force to the environment, the effective position control method is proposed which uses a specialized neuron for friction compensation and a fuzzy neural network for trajectory control. A switch-learning algorithm is introduced for this position controller.

The effectiveness of the proposed hybrid controller is verified by computer simulations with the use of a 3DOF planar robot manipulator model. The results show the significant adaptation ability of the proposed controller against an unknown environment.

References

Fukuda T. and Shibata T. (1991): *Neural network application for robotic motion control.* — J. Robotics and Mechatronics, v.2, No.3, pp.21–25.

Kiguchi K. and Fukuda T.(1995): *Fuzzy neural controller for robot manipulator force control.* — Proc. Joint Conf. of 4th IEEE Int. Conf. *Fuzzy Systems* and 2nd Int. *Fuzzy Engineering Symp. (FUZZ- EEE/IFES'95)*, v.2, pp.869–874.

Kiguchi K. and Fukuda F. (1995): *Robot manipulator contact force control application of fuzzy-neural network.* — Proc. IEEE Int. Conf. *Robotics and Automation*, v.1, pp.875–880.

Kiguchi K. and Necsulescu D. (1993): *Control of multi-DOF robot using neural networks.* — Proc. Workshop *Knowledge-Based Systems & Robotics*, pp.747–754.

Lu Z. and Goldenberg A.A. (1995): *Robot impedance control and force regulation: Theory and Experiments.* — Int. J. Robotics Research, v.14, No.3, pp.225–254.

Luh J.Y.S, Walker W.M. and Paul R.P.C. (1980): *Resolved acceleration control of manipulators.* — IEEE Trans. Automat. Contr., v.AC-25, No.3, pp.468–474.

Mamdani E.H. (1974): *Application of fuzzy algorithms for control of simple dynamical plants.* — Proc. of IEE, v.121, pp.1585–1588.

Popovic M.R., Gorinevsky D.M. and Goldenberg A.A. (1995): *Fuzzy logic controller for accurate positioning of direct-drive mechanism using force pulses.* — Proc. IEEE Int. Conf. *Robotics and Automation*, v.1, pp.1166–1171.

Raibert M.H. and Craig J.J. (1981): *Hybrid position/force control of manipulators.* — Trans. ASME J. Dynamic Systems, Measurement, and Control, v.102, pp.126–133.

Sarkodie-Gyan T. and Willumeit H.P. (1995): *Neuro-Fuzzy Based Autonomous Vehicles.* — Proc. Joint Conf. of 4th IEEE Int. Conf. *Fuzzy Systems* and 2nd Int. *Fuzzy Engineering Symp.* (FUZZ- IEEE/IFES'95), v.2, pp.855–862.

Shibata T., Fukuda T., Kosuge K., Arai F., Tokita M. and Mitsuoka T. (1992): *Hybrid symbolic and neuromorphic control for hierarchical intelligent control.* — Proc. IEEE Int. Conf. *Robotics and Automation*, pp.2081–2086.

Suh I.H.and Kim T.W. (1994): *A fuzzy-neural-network-based visual feedback learning control for robot manipulators.* — Proc. IEEE Int. Conf. *Neural Networks*, v.5, pp.2781–2786.

Yabuta T. and Yamada T. (1990): *Possibility of neural networks controller for robot manipulators.* — Proc. IEEE Int. Conf. *Robotics and Automation*, pp.1686–1691.

Yoshikawa T. (1987): *Dynamic hybrid position/force control of robot manipulators — Description of hand constraints and calculation of joint driving force.* — IEEE J. Robotics and Automation, v.3, No.5, pp.386–392.