Proc. Natl. Sci. Counc. ROC(A)
Vol. 25, No. 6, 2001. pp. 367-376

Neural Network Error Accommodation with Fuzzy Logic Elements in Robot Time Optimal Path Tracking

YAUHENI B. VERYHA

Department of Machine Building
Belarussian State Polytechnical Academy
Minsk, Republic of Belarus

(Received January 5, 2000; Accepted November 22, 2000)

ABSTRACT

Error detection, diagnosis and accommodation play key roles in the operation of autonomous robotic systems. System faults, which typically result in changes of critical system parameters or system dynamics, may lead to degradation in performance. This fact is especially important for time optimal robot control when the system parameters reach their critical values and even small changes can lead to accuracy degradation. This paper investigates the problem of error diagnosis in robotic manipulators under computed torque control using neural network and fuzzy logic elements. A learning architecture with neural networks serving as on-line approximators with fuzzy logic elements in the control unit is used for the diagnosis of robotic system errors and error accommodation. Approximation using neural networks provides a model of the error characteristics that can be used for the detection and elimination of errors in robot functioning. Simulation results illustrate the ability of the neural network based error diagnosis method with the fuzzy elements described in this paper to detect and accommodate errors in a two-link robotic manipulator under time optimal control.

Key Words: time optimal control, adaptive robotic system, autonomous robotic system, error detection, neural network, fuzzy logic element

I. Introduction

Robotic systems are widely used in different complex engineering applications that demand high performance and productivity. Interest in the problem of controlling robotic manipulators in the minimum amount of time is motivated by the desire to reduce the number of cycle times in industrial applications, such as laser cutting, welding, high pressure water jet cutting, etc.

The well-known approach to the problem of time optimal control of robotic manipulators was proposed by Cahill *et al.* (1995). He proposed the theory and two schemes for adapting time optimal trajectory algorithms for application in robots under computed torque control. This provides a practical foundation for achieving near time optimal performance while tracking paths to a prescribed tolerance. In time optimal control of robotic manipulators, the main characteristics of the robotic system (accuracy, cycle time and joint torques) reach their critical values. This means that even a small change in the robotic system parameters may significantly degrade robotic system performance, including engine saturation and accuracy degradation (Bobrow *et al.*, 1985; Freyermuth, 1991).

In application environments, robotic system faults, which are mainly characterized by changes in the system parameters or in the manipulator dynamics, can result not only in loss of productivity, but also in loss of accuracy, as is the case with time optimally controlled robots. Difficult and dangerous environments limit the ability of humans to perform any supervisory or corrective tasks (Ayoubi, 1994; Dawson *et al.*, 1995; Izermann, 1995; Nishivaki, 1986).

This paper presents a learning method based on nonlinear modeling techniques for detecting errors in rigid link robotic manipulators under time optimal control. The robot dynamics are assumed to be known exactly prior to the appearance of errors. The multi-layered neural network is used to monitor the robot for changes in dynamics. With the aid of the fuzzy logic elements, the computational approach to the estimation of robot faults is significantly improved. This results in a decrease in the elapsed time for error accommodation. The approximation capabilities of the neural network are used here not only to detect the occurrence of system failures, but also to provide on-line estimates of the fault characteristics (Antsaklis and Passino, 1992; Ayoubi, 1994; Dash and Panda, 1996).

The main component in the error monitoring architecture is the development of a nonlinear estimation scheme that allows the use of systematic learning procedures for detecting and accommodating errors (Dawson *et al.*, 1995; Driankov *et al.*, 1993; Kandel and Langholz, 1994). The error diagnosis algorithm relies on measurements from sensors used to control the robotic manipulator. Hence, this approach does not require any additional equipment. The process of error diagnosis includes the following stages: detection, diagnosis (identification

of the error cause) and accommodation (reconfiguration of the robotic system to accommodate the error) (Izermann, 1995).

In many applications, error diagnosis schemes are built using hardware redundancy. In these schemes, redundant physical subsystems, for example, multiple sensors, are incorporated into the system. Generally, the additional cost and complexity of the suggested redundant hardware makes these architectures unattractive. Another approach is analytical redundancy. In these architectures, sensor measurements are processed to estimate the value of a desirable variable using a robot model. The estimate is then compared with the measured value of the variable to generate a residual (Freyermuth, 1991; Guez *et al*., 1998).

A number of researchers have worked on the problem of designing automated error diagnosis schemes for robotic systems using analytical redundancy methods, but most of the developed error diagnosis schemes exclusively use linear nominal robotic models with faults that are modeled as external additive input signals of time (Ayoubi, 1994). Accurate representation of the robotic system requires nonlinear modeling. The use of linear techniques results in modeling errors leading in some cases to degraded performance of the error diagnosis system. Robotic system faults often cause unpredictable nonlinear changes in the dynamics of the robot. Thus, to take into account a large class of practical failure situations, a nonlinear modeling framework is required (Izermann, 1995).

II. Robot Dynamics

The robot dynamics can be considered to be governed by

$$\boldsymbol{\tau} = \boldsymbol{M}(\boldsymbol{q})\boldsymbol{q}'' + \boldsymbol{n}(\boldsymbol{q}, \boldsymbol{q}') - \gamma(t - T)\underline{\beta}(\boldsymbol{q}, \boldsymbol{q}', \boldsymbol{\tau}), \qquad (1)$$

where $\boldsymbol{q}$, $\boldsymbol{q}'$ and $\boldsymbol{q}''$ are vectors of joint positions, velocities and accelerations, respectively, $\boldsymbol{\tau}$ is the input torque vector, $\boldsymbol{M}(\boldsymbol{q})$ is the inertia matrix, and $\boldsymbol{n}(\boldsymbol{q}, \boldsymbol{q}')$ represents coriolis, centripetal, friction and gravity terms. The term $\underline{\beta}(\boldsymbol{q}, \boldsymbol{q}', \boldsymbol{\tau})$ is a vector which represents the fault in the robotic manipulator, $\gamma(t - T)$ represents the time profile of the fault, and T is the time of fault occurrence.

For convenience of analysis, the changes in the dynamics due to a fault can be represented as

$$\underline{\beta}(\boldsymbol{q}, \boldsymbol{q}', \boldsymbol{\tau}) = \boldsymbol{M}(\boldsymbol{q})\beta(\boldsymbol{q}, \boldsymbol{q}', \boldsymbol{\tau}). \qquad (2)$$

With this representation of the faults, the robot dynamics can be rewritten as

$$\boldsymbol{q}'' = \boldsymbol{M}^{-1}(\boldsymbol{q})(\boldsymbol{\tau} - \boldsymbol{n}(\boldsymbol{q}, \boldsymbol{q}')) + \gamma(t - T)\beta(\boldsymbol{q}, \boldsymbol{q}', \boldsymbol{\tau}). \qquad (3)$$

Here, β is a function of time and not an explicit function of the position $\boldsymbol{q}$, the velocity $\boldsymbol{q}'$ and the input torque $\boldsymbol{\tau}$.

Robotic manipulator failures are usually characterized by changes in critical system parameters, for example, the mass of a link, or introduction of some unknown structural dynamics, both of which result in nonlinear changes in the manipulator dynamics. Thus, an accurate description of fault conditions resulting in a decrease in robot path tracking performance requires nonlinear modeling of faults. This nonlinear modeling allows deviations β to be nonlinear functions of the positions and velocities of the joints and torque inputs to the links. For example, the change of the inertia matrix $\boldsymbol{M}(\boldsymbol{q})$ to $\underline{\boldsymbol{M}}(\boldsymbol{q})$ can be represented in the formulation described in Eq. (3) by defining β as

$$\beta(\boldsymbol{q}, \boldsymbol{q}', \boldsymbol{\tau}) = (\underline{\boldsymbol{M}}^{-1}(\boldsymbol{q}) - \boldsymbol{M}^{-1}(\boldsymbol{q}))(\boldsymbol{\tau} - \boldsymbol{n}(\boldsymbol{q}, \boldsymbol{q}')). \qquad (4)$$

For a real robot functioning with the proposed error detectioning method, it is necessary to approximate the unknown nonlinear function β on-line. Recent progress in hardware and software implementation has made it possible to use sigmoidal neural networks to approximate and analyze nonlinear models (Izermann, 1995).

The error detection scheme considered in this paper is independent of the type of applied controller used in the robotic system. Here, the computed-torque method is used to obtain a trajectory-tracking controller for the robotic manipulator Eq. (1). The control law is given by

$$\tau = \boldsymbol{M}(\boldsymbol{q})(\boldsymbol{q}_\tau'' + \boldsymbol{k}_v\boldsymbol{e}' + \boldsymbol{k}_p\boldsymbol{e}) + \boldsymbol{n}(\boldsymbol{q}, \boldsymbol{q}'), \qquad (5)$$

where $\boldsymbol{q}_\tau$ is the desired trajectory, $\boldsymbol{e} = \boldsymbol{q}_\tau - \boldsymbol{q}$ is the tracking error, and $\boldsymbol{k}_v$ and $\boldsymbol{k}_p$ are vectors of controller gains.

III. Time Optimal Robot Control

The well-known time optimal trajectory algorithms are related to the following problem (Bobrow, 1988; Bobrow *et al*., 1985). For a given joint space path $\boldsymbol{q} = \boldsymbol{f}(s)$ with the robot dynamics

$$\boldsymbol{M}(\boldsymbol{q})\ \boldsymbol{q}'' + \boldsymbol{n}(\boldsymbol{q}, \boldsymbol{q}') = \boldsymbol{\tau}, \qquad (6)$$

and the actuator constraints

$$\boldsymbol{\tau}_{li}(\boldsymbol{q}, \boldsymbol{q}') \le \boldsymbol{\tau}_i < \boldsymbol{\tau}_{hi}(\boldsymbol{q}, \boldsymbol{q}'), \qquad (7)$$

it is necessary to determine the path-timing $s(t)$; $t \in [0, t_f]$ that minimizes t_f subject to the dynamics in Eq. (6) and actuator constraints in Eq. (7) with $\boldsymbol{q}(t) = \boldsymbol{f}(s(t))$, where $\boldsymbol{q}$ is the vector of joint displacements, s is the path parameter, $\boldsymbol{M}(\boldsymbol{q})$ is the mass matrix, $\boldsymbol{\tau}$ represents the joint torques, and $\boldsymbol{n}(\boldsymbol{q}, \boldsymbol{q}')$ represents coriolis, centripetal, friction and gravity terms. It can be assumed that the path $\boldsymbol{f}(s)$ that is normally specified in task space is given in joint space. This is done to simplify the discussion considering that converting task-space kinematics to joint space path kinematics creates only technical problems

that can easily be handled.

The solutions are based on the following reformulation. The first path constraint $\boldsymbol{q} = \boldsymbol{f}(s)$ is used to eliminate $\boldsymbol{q}$ and its derivatives, yielding dynamics in the form

$$\boldsymbol{A}(s)s'' + \boldsymbol{b}(s, s') = \boldsymbol{\tau}, \quad (8)$$

subject to the following constraints:

$$\boldsymbol{\tau}_{li}(s, s') \le \boldsymbol{\tau}_i \le \boldsymbol{\tau}_{hi}(s, s'), \quad (9)$$

where $\boldsymbol{\tau}_{li}(s, s')$ and $\boldsymbol{\tau}_{hi}(s, s')$ are the lowest and highest values of the compensation torque $\boldsymbol{\tau}_{cm}(s, s')$.

One of the most well-known methods for time optimal trajectory planning was first developed by Bobrow (1988). Later, different improvements were made to this shooting method based on the following. If the admissible control set is convex, then the time-optimal solution will only use controls from the boundary of the admissible set. Based on this principle, shooting methods construct the time-optimal $s(t)$ by means of forward and backward integration in the s-s' plane, using either maximum or minimum acceleration s'' (Hocking, 1991). Thus, the above-mentioned algorithms can be used to determine the time-optimal path timing $s(t)$, which in turn determines the time-optimal joint trajectories $\boldsymbol{q}(t) = \boldsymbol{f}(s(t))$ and torque trajectories $\boldsymbol{\tau}(t)$ (Cahill *et al*., 1995).

In practice, the described algorithms are of limited value because they do not consider the modeling errors (disturbances) or relate these errors to the robot's path-tracking performance. It is known that errors for any single experiment can be represented as disturbances (errors) in joint torques or in joint accelerations (Cahill *et al.*, 1995). Doing so is motivated by the realization that robot path-tracking errors and compensation torques are driven by joint acceleration disturbances through a linear dynamic system:

$$\boldsymbol{e}''(t) + \boldsymbol{k}_v\boldsymbol{e}'(t) + \boldsymbol{k}_p\boldsymbol{e}(t) = \boldsymbol{q}_d''(t) \quad (10)$$

that can be found if the real robot plant is represented as follows:

$$\underline{\boldsymbol{M}}(\boldsymbol{q})\boldsymbol{q}'' + \underline{\boldsymbol{n}}(\boldsymbol{q}, \boldsymbol{q}') + \underline{\boldsymbol{M}}(\boldsymbol{q})\boldsymbol{q}_d''(t) = \boldsymbol{\tau}, \quad (11)$$

where $\underline{\boldsymbol{M}}(\boldsymbol{q})\boldsymbol{q}'' + \underline{\boldsymbol{n}}(\boldsymbol{q}, \boldsymbol{q}')$ is the nominal plant model and $\underline{\boldsymbol{M}}(\boldsymbol{q})\boldsymbol{q}_d''(t)$ accounts for any torques that $\underline{\boldsymbol{M}}(\boldsymbol{q})\boldsymbol{q}'' + \underline{\boldsymbol{n}}(\boldsymbol{q}, \boldsymbol{q}')$ cannot take into account. Thus, $\boldsymbol{q}_d''$ can be interpreted as a joint acceleration disturbance. $\boldsymbol{q}_d''(t)$ for any experiment (Cahill *et al*., 1995) can be identified from on-line measurements of $\boldsymbol{q}$, $\boldsymbol{q}'$, and $\boldsymbol{q}''$ and $\boldsymbol{\tau}$ as follows:

$$\boldsymbol{q}_d'' = \underline{\boldsymbol{M}}^{-1}(\boldsymbol{q})\,(\boldsymbol{\tau} - \underline{\boldsymbol{M}}(\boldsymbol{q})\boldsymbol{q}'' - \underline{\boldsymbol{n}}(\boldsymbol{q}, \boldsymbol{q}')), \quad (12)$$

where $\boldsymbol{\tau}$ represents the torque that is actually applied. Assuming that joint acceleration disturbances remain approximately the same for similar trajectories identified from a trial execution of a trajectory (Cahill *et al*., 1995), they can be used to adjust controller gains to achieve a prescribed tracking accuracy the next time that the trajectory is executed and to predict the compensation torques τ_{cm}, that will be required:

$$\tau = \underline{\boldsymbol{M}}(\boldsymbol{q})\boldsymbol{q}_\tau'' + \underline{\boldsymbol{n}}(\boldsymbol{q}, \boldsymbol{q}') + \tau_{cm}, \quad (13)$$

where

$$\boldsymbol{\tau}_{cm} = \underline{\boldsymbol{M}}(\boldsymbol{q}(t))\,(\boldsymbol{k}_v\boldsymbol{e}'(t) + \boldsymbol{k}_p\boldsymbol{e}(t)), \quad (14)$$

and

$$\boldsymbol{e}(t) = \boldsymbol{q}_\tau(t) - \boldsymbol{q}(t). \quad (15)$$

In order to set the time optimal controller gains, it is necessary to rewrite Eq. (10) in the following way:

$$\Delta\boldsymbol{e}''(t) + \boldsymbol{k}_v\Delta\boldsymbol{e}'(t) + \boldsymbol{k}_p\Delta\boldsymbol{e}(t) + \boldsymbol{k}_p\underline{\boldsymbol{e}} = \boldsymbol{q}_d''(t), \quad (16)$$

where $\underline{\boldsymbol{e}}$ is the required path-tracking accuracy and $\Delta\boldsymbol{e}$ is error fluctuation. Then, error $\boldsymbol{e}(t)$ can be written in the following way:

$$\boldsymbol{e}(t) = \underline{\boldsymbol{e}} + \Delta\boldsymbol{e}(t). \quad (17)$$

From Eq. (16), it is clear that in order to receive the time-optimal controller gains, it is necessary to use the adaptive controller for which the gain $\boldsymbol{k}_p$ must equal

$$\boldsymbol{k}_p = \boldsymbol{q}_d''(t)/\underline{\boldsymbol{e}}. \quad (18)$$

In this case, the dynamic linear system Eq. (16) will set the fluctuation values $\Delta\boldsymbol{e}$ to 0. The $\boldsymbol{k}_v$ gain can be found by using critical damping of the system $\boldsymbol{k}_v = 2\sqrt{\boldsymbol{k}_p}$.

The control system architecture for online trajectory generation in the time optimal robot control is shown in Fig. 1. Controller gains are usually set to their largest stable values

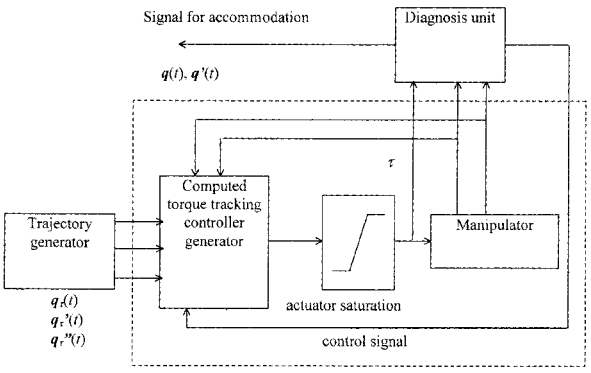

Fig. 1. Control system architecture for online trajectory generation in the time optimal robot control.

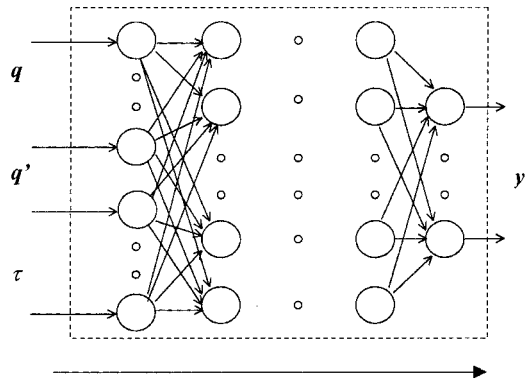

Fig. 2. Multi-layered neural network of the diagnosis unit.

in order to maximize tracking accuracy. But in this case, this principle is not used because large gains will lead to slower time-optimal solutions. Thus, the smallest controller gains as from Eq. (18) that provide sufficient accuracy are used.

The method for choosing controller gains so as to satisfy a prescribed tolerance for tracking accuracy and for predicting the levels of the compensation torque includes the following steps:

(1) identify $\boldsymbol{q}_d''(t)$ via Eq. (12) for a trajectory that is similar to the one considered;
(2) choose controller gains $\boldsymbol{k}_p$ and $\boldsymbol{k}_v$ via Eq. (18);
(3) use Eq. (14) to determine the compensation torque and to estimate the required compensation torque margins $\boldsymbol{\tau}_{lcm}(s, s')$ and $\boldsymbol{\tau}_{hcm}(s, s')$ that bound $\boldsymbol{\tau}_{cm}(s, s')$ (the lowest and highest values of $\boldsymbol{\tau}_{cm}(s, s')$);
(4) calculate a new time-optimal trajectory $\boldsymbol{q}_\tau(t)$ with $\boldsymbol{\tau}_{lcm}(s, s')$ and $\boldsymbol{\tau}_{hcm}(s, s')$ held in reserve:

$$\boldsymbol{\tau}_l(s, s') - \boldsymbol{\tau}_{lcm}(s, s') \le \boldsymbol{\tau} \le \boldsymbol{\tau}_h(s, s') - \boldsymbol{\tau}_{hcm}(s, s'). \quad \textbf{(19)}$$

This trajectory is implemented with the controller gains set to the values chosen at Step 2 of the method. Under the assumption that the acceleration disturbances $\boldsymbol{q}_d''(t)$ will be approximately the same as they were in the trial experiment (Cahill *et al.*, 1995; Ioannou and Sun, 1995), the system driven by this new trajectory should provide time-optimal tracking while satisfying the prescribed tracking error tolerance.

IV. Error Detection Scheme

The error diagnosis scheme for robotic manipulators relies on the following assumptions:

(1) The robotic manipulator has no modeling uncertainties.
(2) In the presence of an error, the states of the robotic system remain bounded.

In practice, assumption 1 cannot be realized due to the fact that the presence of modeling errors will cause a discrepancy between the actual plant and the nominal model, which may result in false alarms. There are different approaches that help to resolve this problem: heuristic tools, threshold value, and decoupling the effects of faults and disturbances (Ayoubi, 1994; Izermann, 1995).

In this work, a small threshold is used in the residual error to account for modeling uncertainties; in this case a fault is declared if the residual error is greater than the selected threshold. The threshold value can be determined on the basis of the example simulation of the robotic system.

As the detection module, the sigmoidal neural network is used. The schematic representation of the multi-layered neural network is shown in Fig. 2.

The multi-layered neural network characteristics can be described by

$$\boldsymbol{y} = \underline{\beta}(\boldsymbol{q}, \boldsymbol{q}', \boldsymbol{\tau}, \underline{\theta}), \quad \textbf{(20)}$$

where $\boldsymbol{q}$, $\boldsymbol{q}'$, and $\boldsymbol{\tau}$ are the inputs to the network, $\boldsymbol{y}$ is the output of the network and $\underline{\theta}$ represents the adjustable weights of the network. The weights $\underline{\theta}(0) = \underline{\theta}_0$ of the selected neural network can be initialized as

$$\underline{\beta}(\boldsymbol{q}, \boldsymbol{q}', \boldsymbol{\tau}, \underline{\theta}_0) = 0, \quad \textbf{(21)}$$

corresponding to the no-error situation. This can be realized by initializing the output weights of the networks to zero. Starting from these initial conditions, the main goal is to calculate, using input and output information, the parameter estimate $\underline{\beta}(t)$ at each time t so that $\underline{\beta}(\boldsymbol{q}, \boldsymbol{q}', \boldsymbol{\tau}, \underline{\theta})$ approximates the unknown function $\gamma(t - T)\underline{\beta}(\boldsymbol{q}, \boldsymbol{q}', \boldsymbol{\tau})$ from Eq. (1). As soon as this is achieved, the output of the neural network $\underline{\beta}$ can be used not only to detect any system errors (failures), but also to set an estimate of the fault β as a function of the inputs $\boldsymbol{q}$, $\boldsymbol{q}'$ and $\boldsymbol{\tau}$.

An error measure between β and $\underline{\beta}$ is needed to update parameter vector $\underline{\theta}$. A suitable error quantity for adjusting the network weights needs to be obtained because β is unknown and unmeasurable. The following estimation model is used to generate an error measure required to update the weights of the neural network:

$$\boldsymbol{\tau} = -\boldsymbol{p}\boldsymbol{M}(\boldsymbol{q})(\boldsymbol{q}' - \omega) + \boldsymbol{M}(\boldsymbol{q})\omega' + \boldsymbol{n}(\boldsymbol{q}, \boldsymbol{q}') - \boldsymbol{M}(\boldsymbol{q})\underline{\beta}(\boldsymbol{q}, \boldsymbol{q}', \boldsymbol{\tau}, \underline{\theta}), \quad \textbf{(22)}$$

where ω is the estimate of the manipulator velocity vector and $\boldsymbol{p}$ is a positive design constant. The estimate $\omega(0)$ is initialized so that $\omega(0) = \boldsymbol{q}'(0)$. The advantage of the presented estimation model is that it can be implemented in the form of a stable filter as follows:

$$\omega' = -\boldsymbol{p}\omega + \boldsymbol{b}(\boldsymbol{q}, \boldsymbol{q}', \boldsymbol{\tau}, \underline{\theta}), \ \omega(0) = \boldsymbol{q}'(0), \quad \textbf{(23)}$$

where ω is the output of the filter $\boldsymbol{w}(s) = 1/(s + \boldsymbol{p})$, with the input

$$\boldsymbol{b}(\boldsymbol{q}, \boldsymbol{q}', \boldsymbol{\tau}, \underline{\theta})$$

$$= \boldsymbol{p}\boldsymbol{q}' + \boldsymbol{M}^{-1}(\boldsymbol{q})(\boldsymbol{\tau} - \boldsymbol{n}(\boldsymbol{q}, \boldsymbol{q}')) + \underline{\beta}(\boldsymbol{q}, \boldsymbol{q}', \boldsymbol{\tau}, \underline{\theta}). \quad \textbf{(24)}$$

The proposed estimation model Eq. (22) is easy to implement and has satisfactory stability and performance properties. The general stability and performance properties for the case of an abrupt failure that occurs at some unknown time instant T were investigated based on the report of Ioannou and Sun (1995). The abrupt fault changes the dynamics of the robot but retains the boundedness of the position and the velocities of the joints. Analysis of the system leads to a conclusion that the overall system remains stable in the presence of a fault (Ioannou and Sun, 1995).

The error between the measured velocity vector and its estimates can he found using $\boldsymbol{E} = \boldsymbol{q}' - \omega$. The following adaptive law can be used to adapt the weights of the neural network for $\underline{\theta}(0) = \underline{\theta}_0$:

$$\dot{\underline{\theta}} = \boldsymbol{G}[\boldsymbol{\Gamma}\boldsymbol{Z}^T\boldsymbol{E}], \quad \textbf{(25)}$$

where $\boldsymbol{\Gamma} = \boldsymbol{\Gamma}^T$ is a positive learning rate matrix, $\boldsymbol{Z} = [\partial\underline{\beta}(\boldsymbol{q}, \boldsymbol{q}', \tau, \underline{\theta}]/\partial\underline{\theta}$ is the gradient matrix of the neural network with respect to the weights, and $\boldsymbol{G}$ is the projection operator. The projection operator restricts the parameter estimate vector $\underline{\theta}$ to some selected compact; for example, a convex region may be used to avoid parameter drift, a phenomenon that may appear with standard adaptive laws in the presence of modeling uncertainties (Dawson *et al.*, 1995). An error is declared whenever the output of the neural network becomes nonzero, which is in general equivalent to the estimation error $\boldsymbol{E}$ becoming nonzero. The way to improve the robustness of the algorithm with respect to modeling uncertainties is to set a threshold value for the fault whenever $|\boldsymbol{E}| \geq \delta$, where δ is a threshold value that depends on the magnitude of the modeling uncertainties.

The main advantage of using the proposed learning methodology in error diagnosis is that it can be used not only to detect the occurrence of an error, but also to provide a model of the fault via the input-output characteristics of the neural network. This error model can be used for failure diagnosis and accommodation by reconfiguring the control law. Automated failure accommodation is considered to be one of the major challenges in designing intelligent robotic systems. One of the nonlinear control tools for controller reconfiguration is feedback linearization (Ayoubi, 1994). The core of this method is transformation of the nonlinear system into a linear one through a change of coordinates and nonlinear feedback. If feedback linearization is achievable, then it is possible to achieve cancellation of the nonlinear functions and the desired closed-loop performance through application of the linear control theory. The post-fault robotic system model can be presented as

$$\boldsymbol{\tau} = \boldsymbol{M}(\boldsymbol{q})\boldsymbol{q}'' + \underline{\boldsymbol{n}}(\boldsymbol{q}, \boldsymbol{q}') + \boldsymbol{M}(\boldsymbol{q})\eta(\boldsymbol{q}, \boldsymbol{q}'; \underline{\theta}), \quad \textbf{(26)}$$

where η is the neural network output. Using the feedback linearization technique, the control law Eq. (5) can be transformed to obtain

$$\boldsymbol{\tau}_\tau = \boldsymbol{\tau} - \boldsymbol{M}(\boldsymbol{q})\eta(\boldsymbol{q}, \boldsymbol{q}'; \underline{\theta}), \quad \textbf{(27)}$$

where $\boldsymbol{\tau}$ is the nominal control law and $\boldsymbol{\tau}_\tau$ is the reconfigured control law. This theory illustrates the ability of the error diagnosis scheme to provide a post failure model that enables accommodation of system failures via the control reconfiguration.

V. Fuzzy Logic Elements

In addition to the error detection method in the robotic system control, it is important to consider the next level of the control unit. Therefore, it is important to discuss the fault (error) tree arising in the system. Fuzzy fault tree analysis has become an efficient tool for improving the reliability, robustness and performance of the overall robotic system (Nishivaki, 1986). A fault tree is a model that represents various logical combinations of possible events that lead to the top event (undesirable event (fault)) of a system.

In this paper, the importance of using the fuzzy fault tree in the analysis of the accommodation signal output is recognized. In the time optimal control with the error detection scheme, one generally can divide faults into two types, first, those errors that occur due to changes in the robot dynamics and other robotic system events, and second, those errors that occur due to actuator saturation in joints. The second fault can be eliminated by repeating appropriate trajectory planning steps (Cahill *et al.*, 1995). Thus, in this case, it is not necessary to use robot reconfiguration as required for the first fault.

To make a suitable fault tree, the following events in the system are defined: A: the accommodation signal; X_1: an error in the robot path-tracking; X_2: actuator saturation; X_3: an error in the robot dynamics and other faults. Then, the top event A (accommodation signal) can be described as

$$A = X_3 \vee (X_1 \wedge X_2). \quad \textbf{(28)}$$

Given the fault probability of X_i and P_{Xi}, the probability of the top event, A can be written as

$$P_A(P_{X1}, P_{X2}, P_{X3}) = 1 - (1 - P_{X3})\,(1 - P_{X1}P_{X2}). \quad \textbf{(29)}$$

The problem considered here is that of calculating the possibility that the top event will appear as a fuzzy set. This is equivalent to determining the following fuzzy set:

$$\underline{P}_A(\underline{P}_{X1}, \underline{P}_{X2}, \underline{P}_{X3}) = 1 - (1 - \underline{P}_{X3})\,(1 - \underline{P}_{X1}\underline{P}_{X2}), \quad \textbf{(30)}$$

where $\underline{P}_{Xi}$ is a fuzzy set defined on [0, 1].

It is often difficult to assign a unique numerical value between 0 and 1 to a fault probability. To overcome this

difficulty, the fault probability can be defined in a certain range on [0,1] and is used here instead of a unique value of the probability. More specifically, the following type of fuzzy set is considered when analysing a fault tree (Nishivaki, 1986). The error probability is defined by

$$\underline{P}_{Xi} = (q_i^L, p_i^L, p_i^R, q_i^R), \tag{31}$$

which is defined by the following function:

$$\underline{P}_{Xi} = \begin{cases} 0, & \text{for } 0 \le p \le q_i^L \\ 1 - \dfrac{p_i^L - p}{\Delta p_i^L}, & \text{for } q_i^L \le p \le p_i^L \\ 1, & \text{for } p_i^L \le p \le p_i^R \\ 1 - \dfrac{p - p_i^R}{\Delta p_i^R}, & \text{for } p_i^R \le p \le q_i^R \\ 0, & \text{for } q_i^R \le p \le 1, \end{cases} \tag{32}$$

where p is the unique value of the event probability, $\Delta p_i^L = p_i^L - q_i^L$ is the difference between the left side nodes of the fuzzy probability, as shown in Fig. 3, and $\Delta p_i^R = q_i^R - p_i^R$ is the difference between the right side nodes of the fuzzy probability. The membership function $\underline{P}_{Xi}(p)$ and the values of the probability q_i^L, p_i^L, p_i^R and q_i^R are shown in Fig. 3.

Taking into account the extension principle, multiplying fuzzy sets $\underline{P}_{Xi}$ $\underline{P}_{Xj}$ gives the following function:

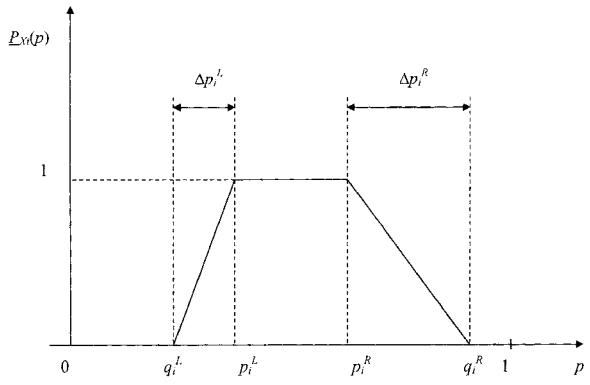

Fig. 3. Membership function $\underline{P}_{Xi}(p)$.

where $h_{i,j}^L = (\Delta p_i^L p_j^L + \Delta p_j^L p_i^L)/(\Delta p_i^L \Delta p_j^L)$ and $h_{i,j}^R = (\Delta p_i^R p_j^R + \Delta p_j^R p_i^R)/(\Delta p_i^R \Delta p_j^R)$ are, accordingly, auxiliary functions used for the compact presentation of $\underline{P}_{XiXj}(p)$.

In order to calculate Eq. (30), the following equation is needed:

$$1 - \underline{P}_{Xi} = (1 - q_j^R, 1 - p_j^R, 1 - p_j^L, 1 - q_j^L). \tag{34}$$

In the following, some definitions are introduced to compare fuzzy probabilities $\underline{P}_{Xi}$ and $\underline{P}_{Xj}$ (Nishivaki, 1986). If $\min(\underline{P}_{Xi}, \underline{P}_{Xj}) = \underline{P}_{Xi}$ or $\max(\underline{P}_{Xi}, \underline{P}_{Xj}) = \underline{P}_{Xj}$, then the order relation is defined as

$$\underline{P}_{Xi} \le \underline{P}_{Xj}. \tag{35}$$

By using the order relation, it is possible to decide which fuzzy probability would be worse. In this case, event X_j should be considered more serious than event X_i. The following order relation exists since the probability of the top event $P_A(p)$ is

$$\underline{P}_{XiXj}(p) = \begin{cases} 0, & \text{for } 0 \le p \le q_i^L q_j^L \\ 1 - \dfrac{h_{ij}^L}{2} + \sqrt{\dfrac{p - p_i^L p_j^L}{\Delta p_i^L \Delta p_j^L} + \left(\dfrac{h_{ij}^L}{2}\right)^2}, & \text{for } q_i^L q_j^L \le p \le p_i^L p_j^L \\ 1, & \text{for } p_i^L p_j^L \le p \le p_i^R p_j^R \\ 1 + \dfrac{h_{ij}^R}{2} - \sqrt{\dfrac{p - p_i^R p_j^R}{\Delta p_i^R \Delta p_j^R} + \left(\dfrac{h_{ij}^R}{2}\right)^2}, & \text{for } p_i^R p_j^R \le p \le q_i^R q_j^R \\ 0, & \text{for } q_i^R q_j^R \le p \le 1, \end{cases} \tag{33}$$

an increasing function with respect to the probability p. It is assumed that Eq. (35) is true; then, the following index $V(\underline{P}_{Xi}, \underline{P}_{Xj})$ for measuring the difference between $\underline{P}_{Xi}$ and $\underline{P}_{Xj}$ is introduced (Nishivaki, 1986):

$$V(\underline{P}_{Xi}, \underline{P}_{Xj}) = (q_j^L - q_i^L) + (p_j^L - p_i^L) + (p_j^R - p_i^R) + (q_j^R - q_i^R). \quad \textbf{(36)}$$

For simplicity, one can define

$$\underline{P}_A(\underline{P}_{X1}, ..., \underline{P}_{Xi}, ..., \underline{P}_{Xn}) = \underline{P}_A, \quad \textbf{(37)}$$

$$\underline{P}_A(\underline{P}_{X1}, ..., \underline{P}_{Xi}, 0, \underline{P}_{Xi+1}, ..., \underline{P}_{Xn}) = \underline{P}_{Ai}. \quad \textbf{(38)}$$

Then, the value $V(\underline{P}_{Ai}, \underline{P}_A)$ indicates the improvement attained by eliminating the possibility of fault X_i. For example, if

$$V(\underline{P}_{Ai}, \underline{P}_A) \geq V(\underline{P}_{Aj}, \underline{P}_A), \quad \textbf{(39)}$$

then it is clear that preventing fault X_i is more efficient than removing fault X_j.

Using the above-mentioned fault tree, the following example is considered:

$$V(\underline{P}_{A1}, \underline{P}_A) = V(\underline{P}_{A2}, \underline{P}_A) = 0.066,$$

$$V(\underline{P}_{A3}, \underline{P}_A) = 0.059.$$

Since $V(\underline{P}_{A1}, \underline{P}_A) = V(\underline{P}_{A2}, \underline{P}_A) \geq V(\underline{P}_{A3}, \underline{P}_A)$, the improvement due to the prevention of the fault of X_1 or X_2 is greater than that for X_3. Thus, if possible, the system should prevent and be ready to accommodate faults X_1 and X_2 instead of X_3 in this case.

Application of this type of fuzzy logic element in the robotic control system increases the overall robotic system reliability and robustness. The time period from the control system response to the failure decreases.

VI. Simulation Results

In order to illustrate the proposed error detection scheme, example simulations were carried out on links 2 and 3 of the PUMA type robot RM-01 (prototype designed by Nokia Co. (Finland) and Granat Co. (Belarus)) with six degrees of freedom. The parameters of the PUMA (programmed universal manipulator) are well known. Accordingly with the time optimal robot control, Steps 1 – 4 described in Section III were completed. The robot model included all the rigid body inertial effects and independent coulumb and viscous friction terms for each joint. Torque limits in the form of Eq. (7) were formed on the base of the current and voltage limits of joint motors (τ_{li} = –90 Nm and τ_{hi} = 90 Nm). More detailed experimental data can be found in Veryha (1999). The acceleration disturbances $\boldsymbol{q}_d$"(t) experimentally identified from one experiment were used later to predict the joint errors $\boldsymbol{e}(t)$ and compensation torques $\tau_{cm}(t)$ for a number of subsequent experiments with different controller gains. For the calculation of $\boldsymbol{q}_d$"(t), signals $\boldsymbol{q}(t)$ and $\boldsymbol{q}$'(t) were obtained by means of finite difference estimation of encoder data. Thus, high levels of noise were introduced into the calculated $\boldsymbol{q}_d$"(t) (Veryha, 1999) using Eq. (12). Therefore, to use $\boldsymbol{q}_d$"(t), the data were passed through a low-pass Kalman filter. Following Veryha (1999), use of the low-pass filter for noise elimination allowed prediction of the tracking error $\boldsymbol{e}(t)$ and compensation torque $\tau_{cm}(t)$ with good accuracy.

The joint space path shown in Figs. 4 and 5 was controlled to the described joints of the PUMA type robot with the requirement that the joint errors $\boldsymbol{e}(t)$ be less than (0.005, 0.0025) radians. The desired trajectories for both links are curved lines with an amplitude of 20 degrees and, accordingly, a period of 10 seconds with the appropriate scale dimensions along the axes. These trajectories are semicircles. The initial values of $\boldsymbol{q}_d$"(t) were obtained by driving the system with a trajectory that was theoretically time optimal (Veryha, 1999). After integrating $\boldsymbol{q}_d$"(t) through Eq. (12) with different gains, the resulting gains $\boldsymbol{k}_p$ = (191.4, 942.2) and $\boldsymbol{k}_v$ = (27.7, 61.4) were those at which the prescribed tolerance was obtained. Thus, the system with this gain setting behaved in the optimal way (the errors were within the given limits $\underline{\boldsymbol{e}}$ and the optimal quickness was obtained). If under these conditions, deviations (faults) appear, then the error accommodation system should be used.

For error detection for two links of the PUMA type robot, a three-layer sigmoidal neural network with 6 neurons in the input layer, 35 neurons in the hidden layer and 3 neurons in

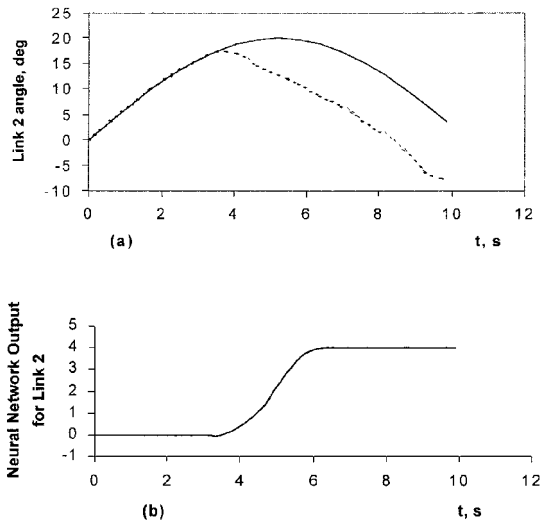

Fig. 4. (a) Joint angle and (b) neural network output for Link 2 of the PUMA manipulator. (— desired trajectory, ---- real trajectory)

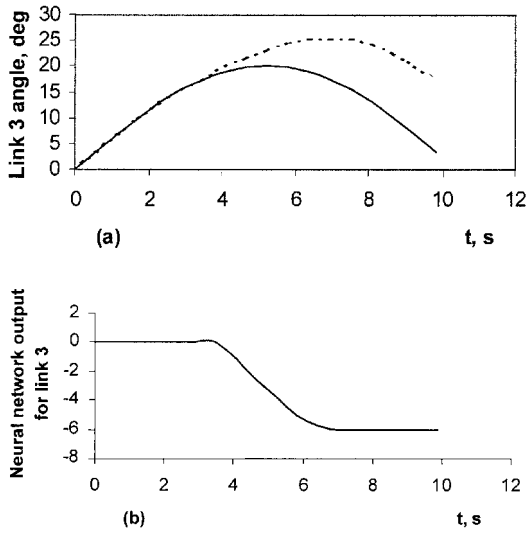

Fig. 5. (a) Joint angle and (b) neural network output for Link 3 of the PUMA manipulator. (— desired trajectory, ---- real trajectory)

the output layer was used. Since the control law Eq. (5) was a function of $\boldsymbol{q}$ and $\boldsymbol{q}$', the neural network was used to estimate the function $\eta(\boldsymbol{q}, \boldsymbol{q}') = \beta(\boldsymbol{q}, \boldsymbol{q}', \tau(\boldsymbol{q}, \boldsymbol{q}'))$ in all the simulations. The inputs to the neural network were the vectors $\boldsymbol{q}$ and $\boldsymbol{q}$'. The filter pole equaled 1, the learning rate $\boldsymbol{\Gamma}$ was set to 2. 6, and the size of the hypershere for the projection algorithm was selected as 12. All the simulations were performed using MathCAD. The robotic system was simulated with an error that occurred at t = 4 seconds, resulting in a 10% change in the mass of a third link of the PUMA robot. This caused deviation in $\boldsymbol{M}(\boldsymbol{q})$ and $\boldsymbol{n}(\boldsymbol{q}, \boldsymbol{q}')$, resulting in a change in the dynamics of the robotic system. Figures 4 and 5 show plots of the joint angles and neural network outputs.

Prior to the occurrence of a fault ($t \le 4$ sec), the control law caused the joint angles to follow the required trajectories; in this case, the outputs of the neural networks were nonzero. As can be seen from Figs. 4 and 5, the fault at t = 4 sec caused a significant tracking error in both links. The neural network outputs went to nonzero values very soon after the fault. This indicated the existence of the fault in the system. Then, the system reconfiguration was used that is clearly explained in Section IV. Figure 6 shows the trajectories of the robot links when the reconfiguration control law was used.

Comparing this figure with Figs. 4 and 5, the following conclusion follows. The trajectory tracking error is considerably reduced in Fig. 6. Therefore, the fault was detected and dealt with by the proposed method in a rather fast way. These results illustrate the ability of the proposed error diagnosis scheme to deal with system failures via control reconfiguration. Nevertheless, there is still room to improve the overall system performance. Based on the simulation results, the value of the time delay for system accommodation (Fig. 6) was calculated. This value was 0.34 sec in the given simulation. In the time optimal control with the error detection scheme, it is possible to define two types of faults: first, those errors that occur due to changes in the robot dynamics and other robotic system events, and second, those errors that occur due to actuator saturation in joints. Here, in the simulated example, the autonomous robotic system always used the reconfiguration law when a fault occurred. In the actual system, a fault of the second type (actuator saturation) can also take place in the robotic system (Cahill *et al*., 1995). In this case, the reconfiguration law is not required. Generally it is not easy in the autonomous system to distinguish a fault. Thus, a further system simulation was performed using the parameter estimation algorithm (Ayoubi, 1994) that allowed to distinguish a fault. This algorithm started with the assumption that the fault was of the first type; then, an appropriate procedure with the reconfiguration law was used. If further system behavior showed that the wrong assumption was wrong, then the second procedure was used (repetition of appropriate trajectory planning steps (Cahill *et al*., 1995)). Generally, in order to set the assumption for first or second fault priority, it is necessary to use appropriate fuzzy sets (Veryha, 1999). First, the above-mentioned simulation (Fig. 6) was performed with the assumption of first fault priority, and the time delay for accommodation was set to 0.34 sec. Then, the simulation was performed with the assumption of second fault priority. The time delay for accommodation in the second case was 0.94 sec. Thus, the time delay increased more than two times. The best way to correctly determine the assumption for robotic system fault accommodation is to use fuzzy logic as described in Section V. On the basis of the statistical and probabilistic information about real equipment, the appropriate fuzzy sets

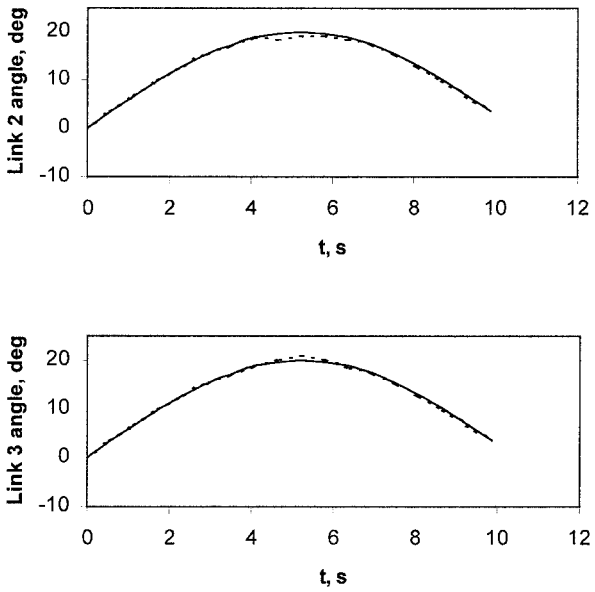

Fig. 6. Trajectories of the PUMA robot links when the reconfiguration control law was used. (— desired trajectory, ---- real trajectory)

can be formed, as discussed in Section V. The formation of appropriate fuzzy sets for the given autonomous robotic system will be conducted in future research on the basis of experimental data for industrial applications applied using the methodology developed here.

VII. Conclusion

In this paper, the problem of error detection and diagnosis in robotic manipulators has been investigated. A learning scheme with neural networks, as approximators of off-nominal robotic system behavior under computed torque control with time optimal path tracking, has been used to monitor the robotic system for faults. Approximation of the off-nominal robot performance has provided a model of the error characteristics that can be used for the detection of faults. The use of fuzzy logic elements in the robotic system control module has been proposed in order to improve the robustness, reliability and performance of the overall robotic system. Simulation results confirm the ability of the neural network based error diagnosis method to detect and accommodate faults in two links of the PUMA manipulator.

This paper has presented an approach to designing nonlinear fault diagnosis algorithms. Future work will focus on more detailed investigation of the system performance properties and fuzzy logic data formation based on actual implementation of the robotic system in industrial enterprises with hazardous environments.

Acknowledgment

This research was supported in part by the Belarussian State Polytechnical Academy under the Cooperative Research Program. The author wishes to express his gratitude to the referees for their valuable comments and to my students who helped me to form computer models of the simulated robotic system with the diagnosis unit.

References

Antsaklis, P. and K. Passino (1992) *An Introduction to Intelligent and Autonomous Control*. Kluwer, Norwell, MA, U.S.A.

Ayoubi, M. (1994) Nonlinear dynamic system identification with dynamic neural structure for fault diagnosis in technical processes. *Proc. IEEE Int'l. Conf. Syst., Man and Cyb*., San Antonio, TX, U.S.A.

Bobrow, J. (1988) Optimal robot path planning using the minimum-time criterion. *IEEE Trans. Robot. Automat.*, **4**(1), 443-450.

Bobrow, J. E., S. Dubowsky, and J. S. Gibson (1985) Time-optimal control of robotic manipulators along specified paths. *Int'l. J. Robotics Res*., **4**(3), 3-17.

Cahill, A. J., J. Kieffer, and M. R. James (1995) On representing robot modeling errors as disturbances in joint accelerations: Theory and experiment. *Proc. IEEE Conf. Decision Contr.*, New Orleans, LA, U.S.A.

Dash, P. K. and S. K. Panda (1996) Gain-scheduling adaptive control strategies for HDVC systems using fuzzy logic. *Proc. Int'l. Conf. Power Electronics, Drives and Energy Systems*, New Delhi, India.

Dawson, D. M., M. M. Bridges, and Z. Qu (1995) *Nonlinear Control of Robotic Systems for Environmental Waste and Restoration*. Prentice Hall, Englewood Cliffs, NJ, U.S.A.

Driankov, D., H. Hellendoorn, and M. Reinfrank (1993) *An Introduction to Fuzzy Control*. Springer, Berlin, Germany.

Freyermuth, B. (1991) An approach to model based fault diagnosis of industrial robots. *Proc. IEEE Int'l. Conf. Robot. Automat*., Sacramento, CA, U.S.A.

Guez, A., J. L. Eilbert, and M. Kam (1998) Neural network architecture for control. *IEEE Contr. Syst. Mag.*, **8**(2), 22-25.

Hocking, L. (1991) *Optimal Control*. Clarendon, Oxford, U.K.

Ioannou, P. A. and J. Sun (1995) *Stable and Robust Adaptive Control*. Prentice-Hall, Englewood Cliffs, NJ, U.S.A.

Izermann, R. (1995) Model based fault detection and diagnosis methods. *Proc. American Control Conf*., Seattle, WA, U.S.A.

Kandel, A. and G. Langholz (1994) *Fuzzy Control Systems*. CRC Press, Boca Raton, FL, U.S.A.

Nishivaki, Y. (1986) Possible application of fuzzy set theory to nuclear safety analysis, risk perception and nuclear plant siting evaluation. *Proc. Int'l. Conf. on Fuzzy Sets Appl*., Berlin, Germany.

Veryha, Y. (1999) Modelling of kinematic errors of robotic assembly manipulators with probabilistic models, *Proc. Int'l. Symp. "Reliability & Quality – 99" of Russian Academy of Science*, Penza, Russia.

Y.B. Veryha

以神經網路與模糊邏輯增進機器人時間最佳化路徑追蹤之容錯度

YAUHENI B. VERYHA

Department of Machine Building
Belarussian State Polytechnical Academy
Minsk, Republic of Belarus

摘　要

錯誤的偵測、診斷、與承受對自主性機器人系統是相當重要的，系統的錯誤常常發生於系統關鍵參數或動態產生變化時，也因此會惡化系統的表現，此種現象對於機器人時間最佳化控制的影響更爲顯著，因爲在此情形下，系統參數常是在關鍵值，而即使是很小的誤差也會導致相當大的錯誤。此篇論文乃利用神經網路以及模糊邏輯來處理在計算力矩控制（computed torque control）下機器人錯誤診斷的問題，所提出的學習架構包含一神經網路作爲即時逼近器（on-line approximator）以及在控制部分的模糊邏輯單元來進行機器人系統的錯誤診斷與容錯。利用神經網路進行的逼近提供了錯誤特徵的模型，可用來偵測與消除機器人功能上的錯誤。模擬結果印證了所提出的學習機制足以處理在時間最佳化控制下兩軸機器人的錯誤偵測與容錯。