# Implementation of a Robust Dynamic Control for SCARA Robot

**Jang Myung Lee*, Min-Cheol Lee**, Kwon Son**, Man Hyung Lee** and Sung Hyun Han*****

*(Received February 11, 1998)*

A control system for SCARA robot is designed for implementing a robust dynamic control algorithm. This study focuses on the use of DSPs in the design of joint controllers and interfaces in between the host controller and four joint controllers and in between the joint controllers and four servo drives. The mechanical body of SCARA robot and the servo drives are selected from the commercially available products. The four joint controllers, assigned to each joint separately, are combined into a common system through the mother board hardwarewise and through the global memory softwarewise. The mother board is designed to connect joint controllers onto the board through the slots adopting PC/104 bus structures. The global memory stores the common data which can be shared by joint controllers and used by the host computer directly, and it virtually combines the whole system into one. To demonstrate the performance and efficiency of the system, a robust inverse dynamic algorithm is proposed and implemented for a faster and more precise control. The robust inverse dynamic algorithm is basically derived from an inverse dynamic algorithm and a PID compensator. Based upon the derived dynamic equations of SCARA robot, the inverse dynamic algorithm is initially implemented with 1 msec of control cycle—0.3 msec is actually used for the control algorithm—in this system. The algorithm is found to be inadequate for the high speed and precision tasks due to inherent modelling errors and time-varying factors. Therefore, a variable PID algorithm is combined with the inverse dynamic algorithm to reinforce robustness of control. Experimental data using the proposed algorithm are presented and compared with the results obtained from the PID and the inverse dynamic algorithms.

**Key Words:** Robust Dynamic Control System, PID, Disturbance, DSP.

## Nomenclature

| | |
|---|---|
| $D(q)$ | : is an $n \times n$ inertia matrix. |
| $C(q, \dot{q})$ | : represents the centrifugal and Coriolis terms. |
| $G(q)$ | : is an $n \times 1$ gravity vector. |
| $N(q, \dot{q})$ | : is equal to $C(q, \dot{q}) + G(q)$. |
| $\tau$ | : is the $n \times 1$ torque vector. |
| $J_m$ | : is an inertia matrix of motor. |
| $B_m$ | : is a friction matrix of motor. |
| $R$ | : is a gear ratio matrix. |
| $\tau_m$ | : represents the torque supplied by the actuator. |
| $u$ | : represents a control input vector. |
| $K_P$ | : is a proportional gain coefficient. |
| $K_I$ | : is an integral gain coefficient. |
| $D_D$ | : is a differential gain coefficient. |
| $K_D$ | : is equal to $q_m{}^d - q_m$. |
| $\eta$ | : is equal to $-[J_m + \overline{D_m}(q_m)]^{-1}\varepsilon$. |
| $\varepsilon$ | : is defined by $\|\ddot{e}_i\|$ and non-diagonal terms of $D_m(q_m)$. |
| $\overline{D_m}(q_m)$ | : is a diagonal submatrix of $D_m(q_m)$. |
| $L$ | : is the acronym of large. |

* Department of Electronics Engineering and Research Institute of Mechanical Technology, Pusan National University, Kumjong-Gu, 609-735, Pusan, Korea
** School of Mechanical Engineering and Research Institute of Mechanical Technology, Pusan National University
*** Department of Mechanical Engineering, Kyungnam University

$M$ : is the acronym of medium.
$H$ : is the acronym of high.
$\mu_E(x)$ : is the input membership function of position error for axes 1, 2, and 4.
$\mu_V(y)$ : is the input membership function of velocity error for axes 1, 2, and 4.
$\mu_{KD}(z)$ : is the output membership function of $K_D$ for axes 1, 2, and 4.
$\mu_{KI}(z)$ : is the output membership function of $K_I$ for axes 1, 2, and 4.
$\mu_{PE}(x)$ : is the input membership function of positive z-directional position error for axis 3.
$\mu_{PV}(y)$ : is the input membership function of positive z-directional velocity error for axis 3.
$\mu_{PKD}(z)$ : is the output membership function of positive z-directional $K_D$ for axis 3.
$\mu_{PKI}(z)$ : is the output membership function of positive z-directional $K_I$ for axis 3.
$\mu_{ME}(x)$ : is the input membership function of negative z-directional position error for axis 3.
$\mu_{MV}(y)$ : is the input membership function of negative z-directional velocity error for axis 3.
$\mu_{MKD}(z)$ : is the output membership function of negative z-directional $K_D$ for axis 3.
$\mu_{MKI}(z)$ : is the output membership function of negative z-directional $K_I$ for axis 3.
$\lambda_i$ : represents weights for the input membership functions.

## 1. Introduction

Robots are being frequently utilized in factories to alleviate workers in laborious and time-consuming tasks. Noticeable improvement in speed and precision makes robots more popular tools in manufacturing processes where highly robust and reliable tasks are required. Many researches have been performed through the computer simulation to find a suitable control algorithm for tasks specified at a high level. As a result, many control algorithms have been developed (Lewis *et al.*, 1993; Spong and Ortega, 1994; Song *et al.*, 1994; Shyu *et al.*, 1996), however, few are implemented in the real system. The main difficulty lies in the extended computing time for complex algorithms in which control periods are longer than those of simpler PID algorithms (Rocco, 1996; Kelly and Salgado, 1994). Practically, a prolonged computing time deteriorates the control performance of the system severely. To overcome this difficulty, we have designed an efficient control system using four TMS320C50 digital signal processors (DSPs) for SCARA robot. This system consists of four major parts: a host controller, four joint controllers, four servo drives and the mechanical body of SCARA robot.

The control system introduced here can perform a lot of calculations required for high-speed control algorithms in real time since DSPs are used as main processors of the joint controllers to reduce the computation time. A fixed point DSP, TMS320C50, is selected for our design, as it provides the maximum value of performance/cost for the controller at this moment (Digital Signal Processing Products, 1993; Digital Signal Processing Products, 1989; Lin *et al.*, 1987). A joint controller is assigned to each joint motor to meet the case that independent joint control should be required for the control algorithm and that heavy computations be required for each joint control. In our design, data for interactions among joint motions are stored and released from the global memory through a 16 bits data bus. A host computer (PC 586) is used for the trajectory planning, and teaching of given tasks as well as the graphic display of task performance. The host computer has serial communication channels to joint controllers to provide auxiliary communication path in emergency cases. For example, when a fatal error happens to a joint controller, the controller can not send messages to the host computer through the 16 bits data bus shared by all of the joint controllers. In this case, a hardware-supported signal will notice the host computer to catch the fatal error through the additional serial communication channels.

The inverse dynamic approach requires and utilizes the detailed description of a robotic system. However, it cannot comply with the estimation error of the inertia matrix and external disturbances. Instead of applying the boundedness assumption to derive VSPID(Chen and Chang, 1996) or to derive accurate compensated inverse dynamics(Shyu *et al.*, 1996), we dynamically adjust the PID parameters using fuzzy rules and a fine tuning scheme.

To demonstrate that the designed system is suitable for the application of dynamic control algorithms involving lots of calculations, experiments are conducted with a SCARA robot and observed experimental results are presented.

## 2. Architecture of Controller

### 2.1 System overview

A schematic diagram of the designed controller is shown in Fig. 1. The system is composed of a host computer (PC 586), four joint controllers, four servo drives, a mother board, actuators, and two power supplies. The mechanical body and servo drives of a SCARA robot are being utilized as the control system. Through the mother board (refer to Fig. 1), the host computer sends position commands to the joint controllers at every 16 msec. With this data transmission it receives actual position and velocity values from the controllers to monitor the control performance in real time. The communication between the host computer and the four controllers is mainly performed through the 16 bits data bus connected to the global memory which is partitioned and used by the memory-map method. Reset and start signals are sent from the host computer to the individual joint controllers and status signals of the controllers are sent back to the host computer through the I/O map I/O interface.

### 2.2 Joint controllers

Joint controllers are designed using fixed-point DSPs (TMS320C50) as shown in Fig. 2. A DSP can perform a single-cycle fixed-point instruction every 50 nsec (20 MIPS) when a 40 MHz crystal is used as clock oscillator. It can address 64k times 16 bits of program and data areas individually, and it has 64k I/O capability. In the DSP, 9k times 16 bits of single-cycle program/data RAM and 2k times 16 bits of single-cycle boot ROM are imbedded inherently. There are also several 32 bits registers that can perform 16 bits multiplications without repetition. Each joint controller uses 32k times 16 bits SRAM as the program memory whose access time is 20 nsec. It also uses 32k times 16 bits SRAM as the data memory individually and shares 32k times 16 bits SRAM as the global memory with others. The memory architecture is determined to arbitrate the memory access conflicts among the DSPs.

The communication path in between each joint controller and the corresponding servo drive is supported by a 34 pin connector as shown in

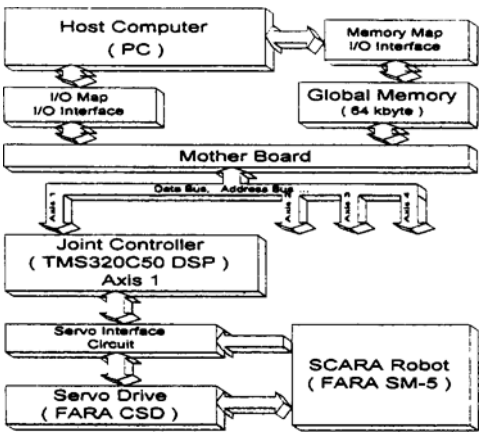



**Fig. 1** Schematic diagram of overall system.

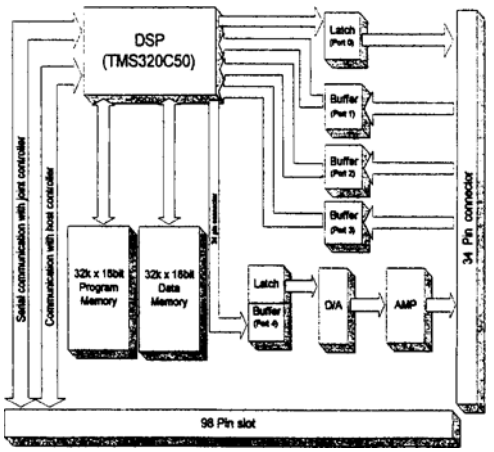



**Fig. 2** Schematic diagram of a joint controller.

**Table 1** Communication signals between a joint controller and a servo drive.

| Address | Bit | Signal | In/Out (DSP) |
|---|---|---|---|
| 50H | 1 | Servo Reset | Output |
| 50H | 1 | Servo On/Off | Output |
| 51H | 3 | Limit Sensor | Input |
| 51H | 4 | Servo Alarm | Input |
| 52H | 16 | Feedback Position | Input |
| | 7 | Reserved, Ground | Input/Output |
| 54H | Analog | Current Command | Output |
| INT3 | 1 | Z-Pulse Interrupt | Input |

Table 1. A 12 bits encoder residing in the servo drive generates the Z phase pulse for the indication of crossing the reference point on the disk of the encoder, and A and B phase pulses for the indication of position and the direction of revolution. The Z phase pulse is noticed to the joint controller by the interrupt, and the A and B phase pulses are fed to the 16 bits up/down counter which is accessed by the joint controller in every control cycle of 1 msec. A servo alarm signal, three alarm codes and a limit sensor signal are also checked by the joint controller; a servo on/off, a reset signal and a current command are sent to the servo drive from the joint controller in every 1 msec as well. Servo setting DIP switches are utilized for the setting of servo drive mode. For example, the servo command can be velocity value or current value depending on the setting of a DIP switch.

The composition of a 34 pin connector and the I/O port addresses of the DSP are summarized in Table 1.

The serial communication network is established among the joint controllers, in addition to the parallel communication path through the global memory, to guarantee the emergency communications among them. Through this serial communication network, all controllers are interconnected. All the communication with the host computer is supported by a 98 pin connector which is plugged onto the mother board.

### 2.3 Servo Drive

Four servo drives (FARACON CSD) in SCARA robot are designed to activate brushless DC motors (BLDC, CSA12, independent power supply type). These BLDC motors have been generally used for fast and accurate control of robotic manipulators (Pillay and Krishnan, 1989; Maeno and Kobata, 1972; Zubek et al. , 1975). Since the driving mechanism of a BLDC is the same as that of an AC servo motor, this motor can be considered as an AC motor.

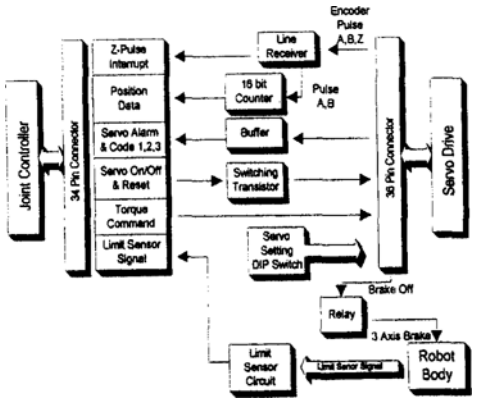



**Fig. 3** Interface in between a joint controller and a servo drive.

As shown in Fig. 3, a 12 bit encoder is attached to each AC servo motor to measure the rotational angle of the motor. The A and B phase pulses of the encoder are fed to a 16 bit up/down counter which are read and cleared by the joint controller in every control cycle; Z-phase signal can be utilized for the initial setting of the robot configuration at home position which is predetermined as well as the reference point on the disk.

### 2.4 Mother board

The mother board configuration is shown in Fig. 4. The board is composed of a global memory, a global memory arbitration logic, serial communication arbitration circuits, EPROM, and several buffers. The global memory consists of 32k times 16 bits SRAM (access time: 20 nsec), and it is mainly used for sending the position commands to joint controllers, which are generated by the host computer as the results of the trajectory planning. Joint controllers can use this global memory to send control results back to the host computer, and can also use it to save and

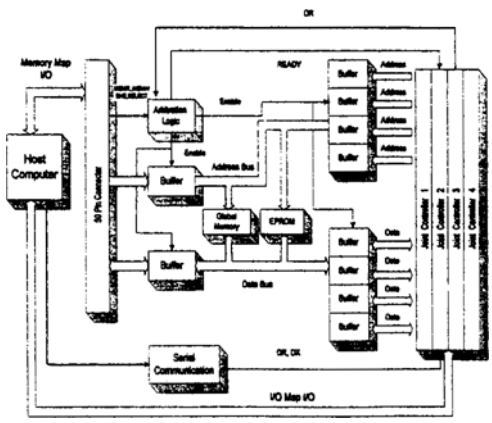



**Fig. 4** Schematic diagram of the mother board.

share the coupling variables among themselves.

The global memory arbitration logic resolves possible collisions which may occur when the global memory is accessed by more than one of the host computer and joint controllers at the same time. In arbitration logic circuits, the order of priority is set by the hardware: the first one is the host computer, and the joint controllers following the order from lower to higher ones. Note that, a lower joint controller needs high priority since coupling effects (which can be accessed through the global memory) in the lower joint are more serious than those in higher joints. The resource allocation, specifically global memory sharing, can be effectively done by closely monitoring coupling variables among joint controllers, even though this effectiveness of the global memory highly depends upon a control algorithm. For example, the sequence of transmitting position commands from the host computer to the joint controllers can be predetermined and kept consistent not to cause collisions.

A serial communication arbitration circuit arbitrates the RS-232C communication (serial communication channels between the host computer and joint controllers) using the multiplexer and demultiplexer. The channel allocation is governed by the host computer and the connection is activated by a polling method. The EPROM is used for the initial down-loading of programs for joint controllers when the power switch is set on.

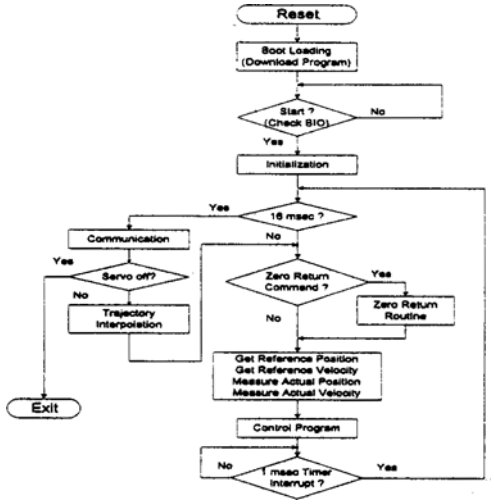



**Fig. 5** Control flowchart of joint controller.

## 3. Control Flow of Joint Controller

A DSP at each joint controller may down-load the main program from the EPROM at the initial stage. In this design, we down-load the main program from the host computer to the joint controllers directly with a selection signal to the joint controllers through the I/O map I/O interface. Note that TMS320C50 can be used in two modes: a micro-computer mode and a microprocessor mode, which can be selected by the bias of $MP/\overline{MC}$ (Microprocessor/Microcomputer mode) pin of the DSP. The micro-computer mode is selected to use the boot loader existing inherently in the DSP. Each DSP waits until all DSPs are down-loaded and a start signal from the host computer comes into the BIO pin of the DSP. After this step, the control flow comes into the normal processing mode as shown in Fig. 5. At the initialization stage of operation, all registers are initialized first and then the initialization of timers for sampling is followed. After the initialization process, the joint controllers start to perform the down-loaded program using the position and velocity data if necessary. The desired position command is received from the host computer in every 16 msec and it is interpolated to 16 segments. Before getting into the servo control loop (16 times of servo control loop in every 1 msec), the joint controller checks whether

the command is for the home position (zero return) or not. And then, the controller generates the reference position and velocity values; it measures the actual position and velocity values. With these values, the control algorithm is performed and the current command is sent to the servo drive in every 1 msec.

## 4. Dynamics of SCARA Robot

The structure of SCARA robot is shown in Fig. 6 with the assignment of link coordinate frames based on the Denavit-Hartenberg representation. The link parameters are listed in Table 2.

The homogeneous transformation matrix, $H_0^4$, and the equations of motion are derived from Table 2 as:

$$H_0^4=\begin{bmatrix} c_{1+2-4} & s_{1+2-4} & 0 & L_1c_1+L_2c_{1+2} \\ s_{1+2-4} & -c_{1+2-4} & 0 & L_1s_1+L_2s_{1+2} \\ 0 & 0 & -1 & d_1+d_3-d_4 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (1)$$

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \\ \dot{\theta} \end{bmatrix}=\begin{bmatrix} L_1s_1-L_2s_{12} & -L_2s_{12} & 0 & 0 \\ L_1c_1+L_2c_{12} & L_2c_{12} & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 1 & 1 & 0 & -1 \end{bmatrix}\begin{bmatrix} \dot{\theta}_1 \\ \dot{\theta}_2 \\ \dot{d}_3 \\ \dot{\theta}_4 \end{bmatrix} \quad (2)$$

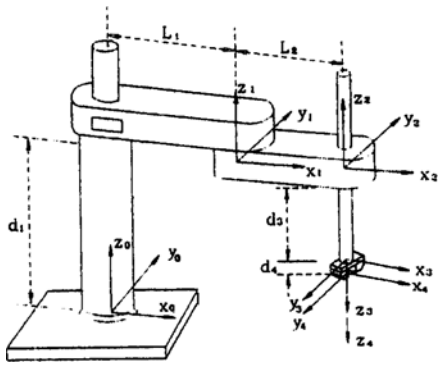



**Fig. 6** SCARA robot.

**Table 2** Link parameters.

| Joint | 1 | 2 | 3 | 4 |
|---|---|---|---|---|
| $a_i$(mm) | 350 | 260 | 0 | 0 |
| $\alpha_i$(°) | 0 | 0 | 180 | 0 |
| $d_i$(mm) | 505 | 0 | $d_3$ | 20 |
| $q_i$ | $q_1$ | $q_2$ | 0 | $q_4$ |

where $c_1=\cos\theta_1$, $s_1=\sin\theta_1$, $c_{1+2-4}=\cos(\theta_1+\theta_2-\theta_4)$, and $s_{12}=\sin\theta_1\sin\theta_2$.

Dynamic equations of SCARA robot are obtained using Euler-Lagrange equations as:

$$\boldsymbol{D}(\boldsymbol{q})\ddot{\boldsymbol{q}}+\boldsymbol{C}(\boldsymbol{q},\ \dot{\boldsymbol{q}})+\dot{\boldsymbol{q}}+\boldsymbol{G}(\boldsymbol{q})=\boldsymbol{\tau} \quad (3)$$

where $\boldsymbol{D}(\boldsymbol{q})$ is an $\boldsymbol{n}\times\boldsymbol{n}$ inertia matrix, $\boldsymbol{D}(\boldsymbol{q})=[\boldsymbol{d}_{ij(q)}]_{n\times n}$, $d_{ij}(q)=d_{ji}(q)$, $\ddot{\boldsymbol{q}}$ is the $n\times1$ acceleration vector, $\boldsymbol{C}(\boldsymbol{q},\ \dot{\boldsymbol{q}})$ represents the centrifugal and Coriolis terms, $\dot{\boldsymbol{q}}$ is an $n\times1$ velocity vector, $\boldsymbol{q}$ is an $n\times1$ configuration vector, $\boldsymbol{G}(\boldsymbol{q})$ is an $n\times1$ gravity vector, and $\boldsymbol{\tau}$ is the $n\times1$ torque vector.

Let $\boldsymbol{N}(\boldsymbol{q},\ \dot{\boldsymbol{q}})=\boldsymbol{C}(\boldsymbol{q},\ \dot{\boldsymbol{q}})+\boldsymbol{G}(\boldsymbol{q})$, and $\boldsymbol{N}(\boldsymbol{q},\ \dot{\boldsymbol{q}})=[\boldsymbol{N}_i(\boldsymbol{q},\ \dot{\boldsymbol{q}})]_{n\times1}$, then $\boldsymbol{D}(\boldsymbol{q})$ and $\boldsymbol{N}(\boldsymbol{q},\ \dot{\boldsymbol{q}})$ can be represented in terms of robot parameters and their elements are obtained as:

$$d_{11}=(I_1+I_2+I_3+I_4)+\left(\frac{m_1}{4}+2m_3+m_4\right)L_1^2+\left(\frac{m_2}{2}+3m_3+m_4\right)L_1L_2\cos q_2+\left(\frac{m_2}{4}+m_3+m_4\right)L_2^2,$$

$$d_{12}=(I_2+I_3+I_4)+\left(\frac{m_2}{4}+m_3+m_4\right)L_2^2+(2m_3+m_4)L_1L_2\cos q_2,$$

$$d_{13}=0,\ d_{14}=I_4,\ d_{22}=(I_2+I_3+I_4)+\left(\frac{m_2}{4}+m_3+m_4\right)L_2^2,\ d_{23}=0,\ d_{24}=I_4,$$

$$d_{33}=m_3+m_4,\ d_{34}=0,\ d_{44}=I_4,$$

$$n_1=\left(\frac{m_2}{2}-m_3\right)L_1L_2\dot{q}_1^2-(4m_3+2m_4)L_1L_2\dot{q}_1\dot{q}_2,$$

$$n_2=\left(\frac{m_2}{2}+m_3+m_4\right)L_1L_2\sin q_2\dot{q}_1^2,$$

$$n_3=(m_3+m_4)g,\ n_4=0,$$

$$m_1=3kg,\ m_2=1.5kg,\ m_3=1.5kg,\ m_4=1.5kg,$$

$$I_1=0.821kgm^2,\ I_2=0.136kgm^2,\ I_3=0.008kgm^2,$$

$$I_4=0.0014kgm^2,\ \text{and}\ g=9.8m/\text{sec}^2.$$

## 5. Robust Inverse Dynamic Algorithm

A PID algorithm is being used in most of industrial robots because it is well developed for the control of linear systems(Takegaki and Arimoto, 1981; Franklin *et al.*, 1986). When the robots are being operated at a low speed, the algorithm shows relatively good performance. In case of a high speed operation, however, the

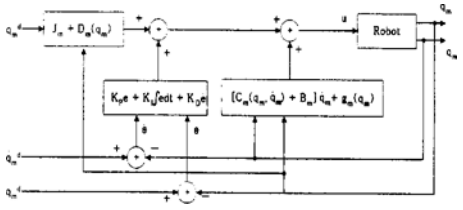



**Fig. 7** Inverse dynamic algorithm.

inertia, centrifugal, Coriolis and gravitational terms are activated and the control performance suddenly drops. To overcome these nonlinear and coupling effects and to make the controller robust against disturbance, we combined an inverse dynamic algorithm and a PID algorithm which is known to be robust against disturbance. Figure 7 shows the control block diagram of the proposed algorithm. Manipulator dynamics can be considered as disturbance to an individual actuator.

Actually the torque $\boldsymbol{\tau}$ is transmitted from an actuator through a reduction gear. Therefore, actuator dynamics including the inertia and viscous friction can be represented as:

$$\boldsymbol{J}_m\ddot{\boldsymbol{q}}+\boldsymbol{B}_m\dot{\boldsymbol{q}}_m=\boldsymbol{\tau}_m-\boldsymbol{R}\boldsymbol{\tau} \tag{4}$$

where $\dot{\boldsymbol{q}}_m$ and $\ddot{\boldsymbol{q}}_m\in R^4$ represent the angular velocity and acceleration, respectively; $\boldsymbol{J}=\text{diag}\{J_{mi},\ i=1,\cdots,4\}$, $\boldsymbol{B}_m=\text{diag}\{B_{mi},\ i=1,\cdots,4\}$, $\boldsymbol{R}=\text{diag}\{r_i,\ i=1,\cdots,4\}$; $r_i$ represents the gear ratio of the $i$-th joint and $\boldsymbol{\tau}_m$ represents the torque supplied by the actuator. Motor's inertia, $\boldsymbol{J}_m$, friction, $\boldsymbol{B}_m$, and gear ratio, $\boldsymbol{R}$, are listed in Table 3.

Plugging of (3) into of (4) for $\boldsymbol{\tau}$ and using $\boldsymbol{q}=R\ \boldsymbol{q}_m$, we have manipulator dynamics (including actuator dynamics) in terms of actuator variables, $\boldsymbol{q}_m$, as (Rocco, 1996):

$$[\boldsymbol{J}_m+\boldsymbol{D}_m(\boldsymbol{q}_m)]\ddot{\boldsymbol{q}}_m+[\boldsymbol{C}_m(\boldsymbol{q}_m,\dot{\boldsymbol{q}}_m)+\boldsymbol{B}_m]\dot{\boldsymbol{q}}_m+\boldsymbol{G}_m(\boldsymbol{q}_m)=\boldsymbol{\tau}_m \tag{5}$$

where $\boldsymbol{D}_m(\boldsymbol{q}_m)=\boldsymbol{R}\ \boldsymbol{D}(\boldsymbol{R}\ \boldsymbol{q}_m)\boldsymbol{R}$, $\boldsymbol{C}_m(\boldsymbol{q}_m,\dot{\boldsymbol{q}}_m)=\boldsymbol{R}\boldsymbol{C}(\boldsymbol{R}\boldsymbol{q}_m,\boldsymbol{R}\dot{\boldsymbol{q}}_m)\boldsymbol{R}$, and $\boldsymbol{G}_m(\boldsymbol{q}_m)=\boldsymbol{R}\ \boldsymbol{G}(\boldsymbol{R}\ \boldsymbol{q}_m)$.

To derive decoupled linear error equations, the nonlinear terms are compensated by using the inverse dynamic approach. To compensate the modelling errors and coupling terms, a PID algorithm is added in the feedback loops as in Fig. 7. As the result, the control law, $\boldsymbol{u}$, is shown as:

**Table 3** Motor's inertia, friction, and gear ratio.

| Joint | 1 | 2 | 3 | 4 |
|---|---|---|---|---|
| $J_{mi}$ (kg · m²) | $0.36\times10^{-4}$ | $0.20\times10^{-4}$ | $0.026\times10^{-4}$ | $0.026\times10^{-4}$ |
| $B_{mi}$ (N · m) | 0.04 | 0.04 | 0.03 | 0.03 |
| $r_i$ | 1/50 | 1/50 | 1/45 | 30.769/360 |

$$\boldsymbol{u}=[\boldsymbol{J}_m+\boldsymbol{D}_m(\boldsymbol{q}_m)]\ddot{\boldsymbol{q}}_m^d+[\boldsymbol{C}_m(\boldsymbol{q}_m,\dot{\boldsymbol{q}}_m)+\boldsymbol{B}_m]\dot{\boldsymbol{q}}_m+\boldsymbol{G}_m(\boldsymbol{q}_m)+\boldsymbol{K}_P\ \boldsymbol{e}+\boldsymbol{K}_I\int\boldsymbol{e}\ dt+\boldsymbol{K}_D\ \dot{\boldsymbol{e}} \tag{6}$$

where $\boldsymbol{K}_P=\text{diag}\{K_{P1},\cdots,K_{P4}\}$, $\boldsymbol{K}_I=\text{diag}\{K_{I1},\cdots,K_{I4}\}$, $\boldsymbol{K}_D=\text{diag}\{K_{D1},\cdots,K_{D4}\}$, and $\boldsymbol{e}=\boldsymbol{q}_m^d-\boldsymbol{q}_m$.

Letting the control law $\boldsymbol{u}$ and the torque $\boldsymbol{\tau}_m$ be equal, the linear error equation can be obtained from (5) and (6) as:

$$[\boldsymbol{J}_m+\boldsymbol{D}_m(\boldsymbol{q}_m)]\ddot{\boldsymbol{e}}+\boldsymbol{K}_P\ \boldsymbol{e}+\boldsymbol{K}_I\int\boldsymbol{e}\ dt+\boldsymbol{K}_D\ \dot{\boldsymbol{e}}=0. \tag{7}$$

Assuming $\ddot{\boldsymbol{q}}_m=\ddot{\boldsymbol{q}}_m^d$ temporarily, that is $\ddot{\boldsymbol{e}}=0$, the equation is simplified as

$$\boldsymbol{K}_P\ \boldsymbol{e}+\boldsymbol{K}_I\int\boldsymbol{e}\ dt+\boldsymbol{K}_D\dot{\boldsymbol{e}}=0. \tag{8}$$

Since $\boldsymbol{K}_P$, $\boldsymbol{K}_I$, and $\boldsymbol{K}_D$ are diagonal matrices in this error equation, the error equation for each actuator are decoupled; they are second order differential equations which can be only represented in terms of actuator variables, $\boldsymbol{q}_{mi}$ and $\boldsymbol{q}_{mi}^d$. Therefore, if we adjust $K_{Pi}$, $K_{Ii}$, and $K_{Di}$ appropriately following the design requirements, we can always maintain the system stable.

Even though the PID compensation (with fixed optimal gains) is included in the inverse dynamic algorithm, the error Eq. (8) can not be satisfied exactly in real time because of time varying modelling errors, such as, friction loss and disturbance. Therefore, in this section a variable PID compensator is designed using fuzzy rules to keep the end-effector of the manipulator on a desired trajectory precisely with the independent control of actuators.

In (7), instead of assuming $\ddot{\boldsymbol{e}}=0$ and deriving (8), we assume $\|\ddot{\boldsymbol{e}}\|=\varepsilon$. Then (7) can be transformed as

$$K_P e + K_I \int e\, dt + K_D \dot{e} = \eta \tag{9}$$

where $\eta = -[J_m + \overline{D_m}(q_m)]$, $\varepsilon = [\varepsilon_1\ \varepsilon_2\ \varepsilon_3\ \varepsilon_4]^T$, and $\overline{D_m}(q_m)$ is a diagonal submatrix of $D_m(q_m)$. Note that $J_m + \overline{D_m}(q_m)$ is a diagonal matrix and the non-diagonal terms of $D_m(q_m)$ are incorporated into $\varepsilon$.

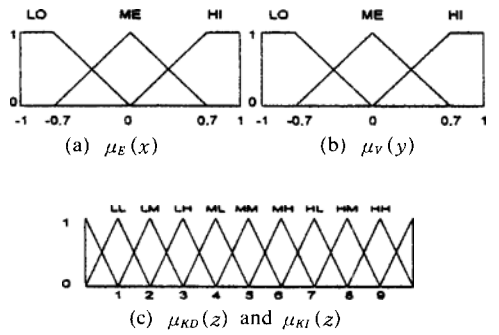
### 5.1 Fuzzy control Rules

Conceptually, a fuzzy control system is a rule-based expert system where expert rules are derived through the experiments. For the robustness of inverse dynamic algorithm in this research, the values of $K_P$, $K_D$ and $K_I$ are tuned in real time based upon the following assumptions: The PD compensator is effective for the improvement of transient behavior, while a PI compensator is valuable for the steady state performance. Our claim is clear for a second order system and the idea is applied to the VSPID(Chen and Chang, 1996). During the run-time, the errors of position and velocity are usually large in the beginning state and become small in a steady state. Therefore, the position and velocity errors are used as input variables for a fuzzy logic control. For the prismatic axis, axis 3, the direction of motion is added to fuzzy input variables since the gravity has primary effect on the motion. This algorithm is implemented through the tuning of PID parameters using the fuzzy rules derived from the experts' experience and knowledge. In this study, the max-min method proposed by Mamdani (1977) is used for fuzzy implication and the mean-of-centers method(Chen and Chang, 1996) is used for defuzzification of fuzzy outputs to obtain the optimal values of $K_D$ and $K_I$ at every sampling instant.

Nine fuzzy control rules are used for the tuning of PID parameters and they are represented as

*If x (position error) is (either L, M, or H) and y (velocity error) is (either L, M, or H), then z (fuzzy output) is (either LL, LM, LH, ···, or HH).*

To obtain PID parameters from fuzzy outputs for axes 1, 2, and 4, let us define membership functions for fuzzy input and output variables as $\mu_E(x)$, $\mu_V(y)$, and $\mu_{KD}(z)$ as shown in Fig. 8.



(a) $\mu_E(x)$ (b) $\mu_V(y)$

(c) $\mu_{KD}(z)$ and $\mu_{KI}(z)$

**Fig. 8** Input/output membership functions for axes 1, 2, and 4.

Then firing strength that is the fitness of the pre-condition part can be represented as(Mamdani, 1977)

$$\lambda i = \mu_{Ei}(x_0) \wedge \mu_{Vi}(y_0) \tag{10}$$

where $\wedge$ represents minimum operation which is a kind of triangular norm corresponding to the conjunctive AND.

The result of fuzzy inference for the nine fuzzy rules following Mamdani's minimum fuzzy implication is given by

$$\mu_{KD}(z_0) = \bigcup_{i=1}^{9} \{\lambda_i \wedge \mu_{KDi}(z_0)\} \tag{11}$$

where $\bigcup$ represents maximum operation which is a triangular-conorm corresponding to the conjunctive OR.

Membership functions for axis 3 are shown in Fig. 9 where $\mu_{PE}(x)$, $\mu_{PV}(y)$, $\mu_{PKD}(z)$, and $\mu_{PKI}(z)$ are for the positive z-directional motion; where $\mu_{ME}(x)$, $\mu_{MV}(y)$, $\mu_{MKD}(z)$, and $\mu_{MKI}(z)$ for the negative motion. Note that four among the nine fuzzy rules are always applied to the given position and velocity errors at a given sampling instance.

The result obtained from (11) are defuzzyfied to obtain crisp PID parameters using the mean-of-centers method as follows

$$K_{Di}(\text{or } K_{Ii}) = \frac{\lambda_1 y_1 + \lambda_2 y_2 + \lambda_3 y_3 + \lambda_4 y_4}{\lambda_1 + \lambda_2 + \lambda_3 + \lambda_4} \tag{12}$$

where $\lambda_1$ to $\lambda_4$ represent weights for the input membership functions selected by (10), and $y_i = \mu K_D^{-1}(\lambda_i)$ $(i = 1, 2, 3, 4)$ represents outputs inferred.

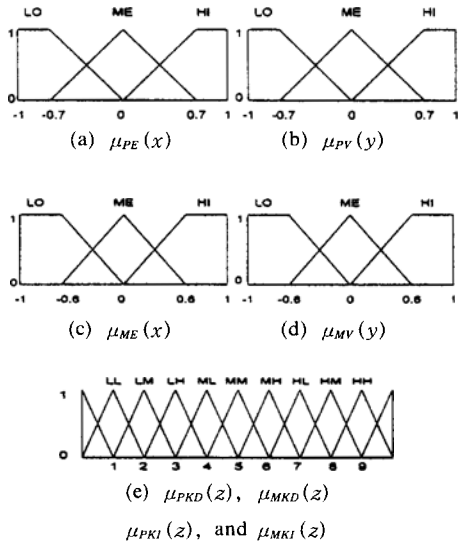



**Fig. 9** Input/output membership functions for axis 3.

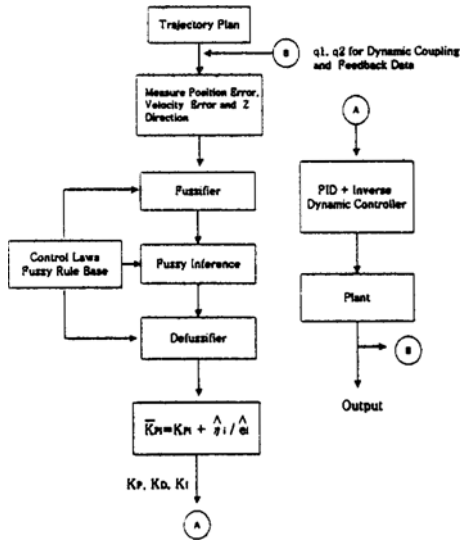



**Fig. 10** Control flowchart of robust inverse dynamic algorithm.

### 5.2 Fine tuning for the compensation of the disturbance

$K_{Ii}$ and $K_{Di}$ are determined through the fuzzy implication, while $K_{Pi}$ is determined to satisfy the stability condition and to meet error characteristics. When the designed PID parameters are applied to the actuator control, (9) representing the error characteristics becomes

$$K_P\hat{e}+K_I\int\hat{e}dt+K_D\,\dot{\hat{e}}=\hat{\eta} \tag{13}$$

where $\hat{}$ stands for the measured value of a variable at each sampling instant.

Note that the error characteristics in (13) is not the same as the desired second order characteristics implied by the designed values of $K_{Pi}$, $K_{Ii}$ and $K_{Di}$ of the non-zero right term. Therefore, $K_{Pi}$ is fine tuned for the compensation of the unexpected disturbance assuming $\dot{\eta}_i=0$ as follows:

$$\overline{K_{Pi}}=K_{Pi}+\frac{\hat{\eta}}{\hat{e}_i} \tag{14}$$

After this fine tuning process, the error equation becomes the second order homogeneous one such that stability is easily guaranteed satisfying the designed error characteristics.

Control flow of the robust inverse dynamic algorithm is shown in Fig. 10 where the processes are represented in terms of individual actuators. The whole implementation of the inverse dynamic algorithm is not indicated directly, however, it should be noted that the communication between a host computer and joint controllers is performed with the aids of global memories which are placed on the mother board for sharing the joint variables.

## 6. Experiment and Controller Evaluation

To check the validity of the controller with the performance evaluation, a conventional PID, an inverse dynamic, and a robust inverse dynamic algorithms are implemented in the SCARA robot. The experimental trajectory is shown in Fig. 11. Velocity profiles for the desired three dimensional straight line motion is shown in Figs. 12 and 13. For a lower speed motion, the end-effector moves from A to B in 0. 88 sec; for a higher speed motion, 0. 72 sec.

For PID algorithm, the optimal parameters are selected during the experiments. For the inverse dynamic and robust inverse dynamic algorithms, the angles and angular velocities of the actuators are stored in the global memory for calculating the manipulator dynamics. For the inverse dynamic algorithm, PID compensation loops are

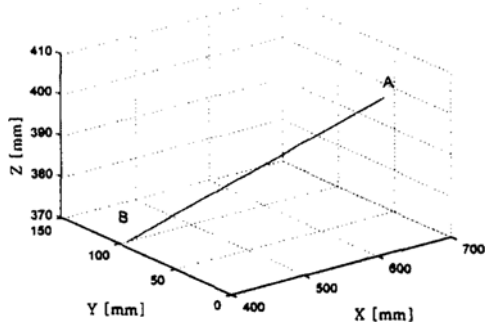



**Fig. 11** Experimental trajectory.

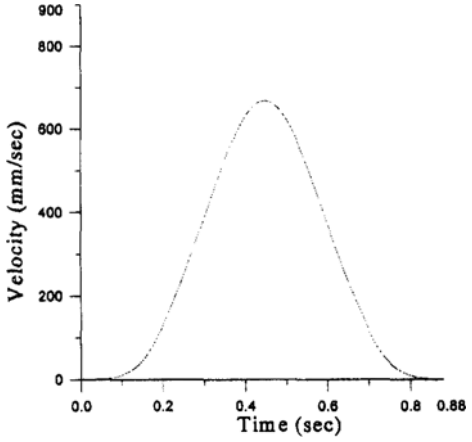



**Fig. 12** Velocity profile for lower speed motion.

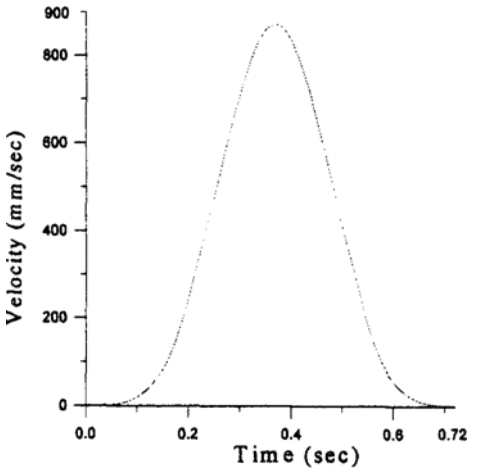



**Fig. 13** Velocity profile for higher speed.

included and the parameters are adjusted to minimize the tracking error through the experiments (refer to Table 4).

For the robust inverse dynamic algorithm, the initial values of PID gains are set to the values used for the inverse dynamic algorithm. And at every control cycle, the gains of $K_{Ii}$ and $K_{Di}$ are adjusted by fuzzy rules as well as the gain of $K_{Pi}$ is finely tuned to drive the error characteristics to follow the second order homogeneous equation. The experimental data are represented both in terms of position error defined as $\sqrt{\delta x^2+\delta y^2+\delta z^2}$ and in terms of velocity error, where $\delta x$, $\delta y$, and $\delta z$ represents $x$, $y$, and $z$ directional position error, respectively. To evaluate the performance of three control algorithms through a comparison, no-load low speed motion and loaded high speed motion are used for the same path as a task for the SCARA robot. Experimental results are shown in Figs. 14 to 17.

Table 4 Optimal PID parameters.

| Gain | PID | Inverse Dynamic |
|---|---|---|
| $K_{P1}$ | 22 | 20 |
| $K_{P2}$ | 13 | 15 |
| $K_{P3}$ | 2 | 2 |
| $K_{P4}$ | 2 | 2 |
| $K_{I1}$ | 0.0032 | 0.0029 |
| $K_{I2}$ | 0.0015 | 0.0013 |
| $K_{I3}$ | 0.0015 | 0.0015 |
| $K_{I4}$ | 0.0012 | 0.0012 |
| $K_{D1}$ | 64 | 61 |
| $K_{D2}$ | 48 | 45 |
| $K_{D3}$ | 48 | 32 |
| $K_{D4}$ | 32 | 32 |

For the case of no-load low speed motion, the inverse dynamic algorithm does not show much better performance than the conventional PID algorithm as shown in Figs. 14 and 15. This result is very convincing, since in a low speed motion, the manipulator dynamics are not a major disturbance source for the SCARA robot using reduction gears. However, the robust inverse dynamic algorithm has showed a great improvement through the dynamic adjustments of PID gains considering the magnitude and derivative of errors. Especially in the region of 0.1 to 0.2 sec where velocity of the motion is abruptly changing, the robust inverse dynamic algorithm

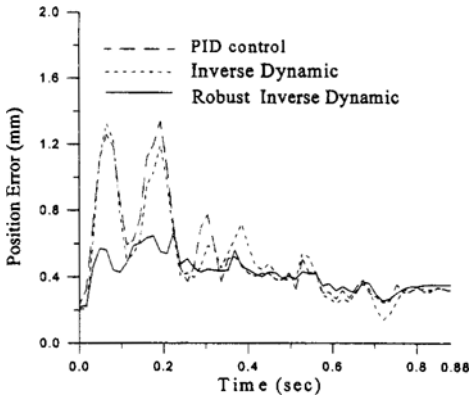



**Fig. 14** Time history of position.

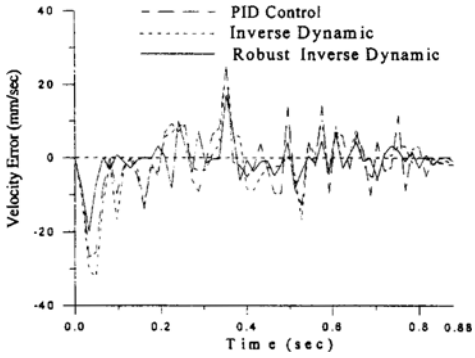



**Fig. 15** Time history of velocity error.

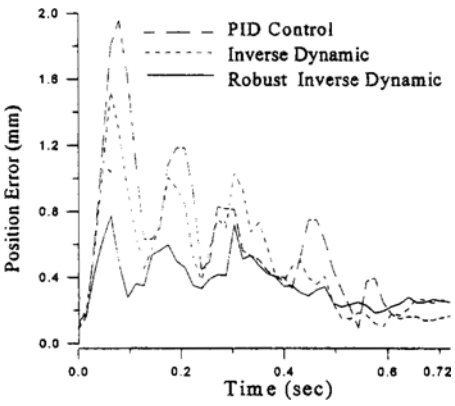



**Fig. 16** Time history of position error.

improves the performance significantly compared with PID and inverse dynamic algorithms. Velocity errors in the robust inverse dynamic algorithm are also much smaller than those in the others.

For the case of high speed motion with a 2 kg load, the inverse dynamic algorithm shows better

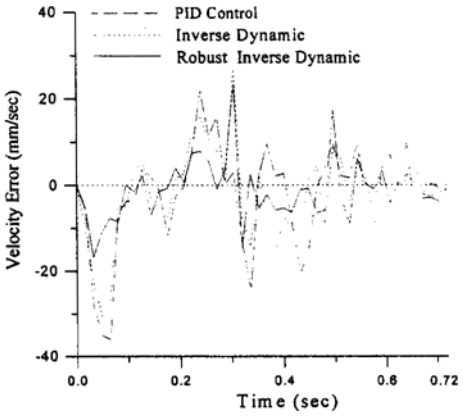



**Fig. 17** Time history of velocity error.

performance than PID algorithm as it is shown in Figs. 16 and 17. As expected, the robust inverse dynamic algorithm showed the best performance for this case as well.

## 7. Conclusions

We have proposed a fast control system for SCARA robot, which is designed using DSPs to implement complex control algorithms requiring many calculations in real time. Four controller boards are used for control of each joint motion independently and quickly, and they are all connected to the host computer through the local bus and the global memory. In this system, both parallel and serial communication networks are provided between the host computer and joint controllers to broaden the application field. To demonstrate the capability and solidness of the designed system as well as to compare the performance of control algorithms, we implemented PID, an inverse dynamic and a robust inverse dynamic control algorithms in this system for a trajectory following task. It is shown that the designed controller can be used to implement different dynamic control algorithms in real time. Our results encourage us to implement various control algorithms in the SCARA robot so that we may develop and implement a high-speed and high-precision control algorithm that utilizes full computing capability. As a preliminary research, a robust dynamic algorithm is proposed and

implemented. Through the experiments, it is shown that the proposed algorithm can give us a solution to the compensation of unmodelled dynamics and disturbance. Developing an intelligent algorithm for self-tuning the PID parameters in the robust dynamic control is left for our future research.

## References

Chen, C. L. and Chang, F. Y., 1996, "Design and Analysis of Neural/Fuzzy Variable Structural PID Control System," *IEE Proc. Control Theory Appl.,* Vol. 143, No. 2, pp. 200~208.

Digital Signal Processing Products, 1989, *Digital Signal Processing Applications with the TMS320 Family,* Vol. 1, 2, 3, Texas Instruments Inc. .

Digital Signal Processing Products, 1993, *TMS320C5X User's Guide,* Texas Instruments Inc. .

Franklin, G. F., Powell, J. D., and Enami -Naeini, A., 1986, *Feedback Control of Dynamic System,* Addison-Wesley.

Kelly, R. and Salgado, R., 1994, "PD Control with Computed Feedforward of Robot Manipulators: A Design Procedure," *IEEE Trans. Robo. and Auto.,* Vol. 10, No. 4.

Lewis, F. L., Abdallak, C. T., and Dawson, D. M., 1993, *Control of Robot Manipulators,* Macmillan Publishing Company.

Lin, K. S., Frantz, G. A., and Simar, R., Jr., 1987, "The TMS320 Family of Digital Signal Processors," *IEEE Proc.,* Vol. 75, No. 9.

Maeno, T., and Kobata, M., 1972, "AC Commutatorless and Brushless Motor," *IEEE Trans. Power Appl. Syst.,* Vol. PAS-91, pp. 1476~1484.

Mamdani, E. H., 1977, "Application of Fuzzy Logic to Approximate Reasoning Using Linguistic Systems," *IEEE Trans. Com.* , C-26, pp. 1182 ~1191.

Pillay, P., and Krishnan, R., 1989, "Modeling, Simulation, and Analysis of Permanent- Magnet Motor Drives, Part II: The Brushless DC Motor Drive," *IEEE Trans. Ind. Appl.,* Vol. 25, No. 2.

Rocco, P., 1996, "Stability of PID Control for Industrial Robot Arms," *IEEE Trans. Robo. and Auto.,* Vol. 12, No. 4.

Shyu, K. K., Chu, P. H., and Shang, L. J., 1996, "Control of Rigid Robot Manipulators via Combination of Adaptive Sliding Mode Control and Compensated Inverse Dynamics Approach," *IEE Proc. Control Theory Appl.,* Vol. 143, No. 3.

Song, Y. D., Mitchell, T. L., and Lai, H. Y., 1994, "Control of a Class of Nonlinear Uncertain Systems via Compensated Inverse Dynamics Approach," *IEEE Trans.,* AC-39, pp. 1866 ~1871.

Spong, M. W., and Ortega, R., 1994, "On Adaptive Inverse Dynamics Control of Rigid Robots," *IEEE Trans.,* AC-39, pp. 1866~1871.

Takegaki, M., and Arimoto, S., 1981, "A New Feedback Method for Dynamic Control of Manipulator," *ASME Trans. of Dynamic System, Measurement, and Control,* Vol. 103, pp. 119~125.

Zubek, J., Abbondanti, A., and Nordby, C. J., 1975, "Pulsewidth Modulated Inverter Motor Drives with Improved Modulation," *IEEE Trans. Ind. Appl.,* Vol. IA-11, pp. 695~703.