Open Access

Simulation Technique of Velocity-Based Discrete-time Control System with Intelligent Control Concepts for Open Architectural Industrial Robots

Fusaomi Nagata*

Department of Electronics and Computer Science, Tokyo University of Science, Yamaguchi

Abstract: Industrial robots have drastically rationalized many kinds of manufacturing processes in industrial fields. For this decade, open architectural industrial robots have been produced from several industrial robot makers such as KAWASAKI Heavy Industries, Ltd., MITSUBISHI Heavy Industries, Ltd. and YASKAWA Electric Corp. and so on. Open architecture described in this article means that the servo system and kinematics of the robot are technically opened, so that various applications required in industrial fields are allowed to be planned and developed at the user side. For example, non-taught operation by using a CAD/CAM system can be considered due to the opened accurate kinematics. Also, force control strategy using a force sensor can be implemented due to the opened servo system. In this article, a simulation technique of velocity-based discrete-time control system for open architectural industrial robots is introduced by giving examples of intelligent control methods. In order to develop a novel velocity-based discrete-time control system for an open architectural industrial robot, it is required from the points of view concerning safety, cost and easiness to preliminarily examine and evaluate the characteristics and performance. In such a case, the proposed simulation technique is useful.

INTRODUCTION

Industrial robots have drastically rationalized many kinds of manufacturing processes in industrial fields. The user interface provided by the robot maker has been almost limited to so-called the teaching pendant. The teaching pendant is a useful and safe tool to obtain positions and orientations at the tip of a robot along a desired trajectory, but the teaching task is very complicated and time-consuming task. Especially, when the target trajectory is a free curved line, many through points must be given to acquire a smooth trajectory; the task is further not easy.

For this decade, open architectural industrial robots as shown in Fig. (1) have been produced from several industrial robot makers such as KAWASAKI Heavy Industries, Ltd., MITSUBISHI Heavy Industries, Ltd. and YASKAWA Electric Corp. and so on. Open architecture described in this article means that the servo system and kinematics of the robot are technically opened, so that various applications required in industrial fields are allowed to be planned and developed at the user side. For example, non-taught operation by using a CAD/CAM system can be considered due to the opened accurate kinematics. Also, force control strategy using a force sensor can be easily implemented due to the technically opened discrete-time servo system.

It is now possible to model and simulate many types of robots. For example, Chen et al. presented a new design of an environment for simulation, animation, and visualization of sensor-driven robots. Although conventional computer-graphics-based robot simulation and animation software packages lacked of capabilities for robot sensing simulation, the system was designed to overcome the deficiency [1]. Also, Benimeli et al. addressed the implementation and comparison of an indirect and a direct identification procedures on an industrial robot provided with an open control architecture. The estimation of dynamic parameters in mechanical systems constituted an issue of crucial importance for dynamic simulation applications where high accuracy was required [2].

In this article, we present a simulation technique of velocity-based discrete-time control system for open architectural industrial robots by giving and combining examples of intelligent control concepts such as genetic algorithms, fuzzy control and neural network. In order to develop a novel velocity-based control system represented in discrete-time domain for an open architectural industrial robot, it is required from the points of view concerning safety, cost and easiness to preliminarily examine and evaluate the characteristics and performance. In such a case, the proposed simulation techniques will be useful. The validation and promise are evaluated through simulations by using a dynamic model of PUMA560 manipulator as shown in Fig. (2) [3], [4].

BASIC SERVO SYSTEM

In order to simulate an industrial robot, first of all, a servo system is considered and designed. Here, the resolved acceleration controller [5] is picked up in a servo system. The resolved acceleration control method or computed torque control method is used for nonlinear control of industrial

Address corresponding to this author at the Department of Electronics and Computer Science, Tokyo University of Science, Yamaguchi;
E-mail: nagata@ed.yamatus.ac.jp

Fig. (1). Open architectural industrial robots.

manipulators, which is composed of a model base portion and a servo portion. The servo portion is a close loop with respect to the position and velocity. On the other hand, the model base portion has the inertia term, gravity term and centrifugal/Coriolis term, which work for canceling the nonlinearity of manipulator. In order to realize high control stability, the position and velocity feedback gains used in the servo portion should be selected suitably.

In this section, a simple but effective fine tuning method after manual tuning process is introduced for the position and velocity feedback gains in the servo portion. At the first step, search space for the gains is roughly narrowed down by a controller designer, e.g. considering the critically damped condition. At the second step, the gains are finely tuned by using genetic algorithms. Genetic algorithms search for a better combination of the position and velocity feedback gains.

Resolved Acceleration Control

The dynamic model of a manipulator without friction term is generally given by

$$\boldsymbol{M}(\boldsymbol{\theta})\ddot{\boldsymbol{\theta}} + \boldsymbol{H}(\boldsymbol{\theta}, \dot{\boldsymbol{\theta}}) + \boldsymbol{G}(\boldsymbol{\theta}) = \boldsymbol{\tau} \tag{1}$$

where, $\boldsymbol{M}(\boldsymbol{\theta}) \in \Re^{6\times 6}$ is the inertia term in joint space. $\boldsymbol{H}(\boldsymbol{\theta}, \dot{\boldsymbol{\theta}}) \in \Re^{6\times 1}$ and $\boldsymbol{G}(\boldsymbol{\theta}) \in \Re^{6\times 1}$ are the Coriolis/centrifugal term and gravity term in joint space, respectively. $\boldsymbol{\theta} \in \Re^{6\times 1}$, $\dot{\boldsymbol{\theta}} \in \Re^{6\times 1}$ and $\ddot{\boldsymbol{\theta}} \in \Re^{6\times 1}$ are the position, velocity and acceleration vectors in joint coordinate system, respectively. $\boldsymbol{\tau} \in \Re^{6\times 1}$ is the joint driving torque vector. In the case that the resolved acceleration control law is employed in the servo system of a manipulator, desired position, velocity and acceleration vectors in Cartesian coordinate system are respectively given to the references of the servo system, so that the joint driving torque is calculated from

$$\begin{aligned} \boldsymbol{\tau} = \hat{\boldsymbol{M}}(\boldsymbol{\theta})\boldsymbol{J}^{-1}(\boldsymbol{\theta}) \times \\ \left[\ddot{\boldsymbol{x}}_r + \boldsymbol{K}_v\{\dot{\boldsymbol{x}}_r - \dot{\boldsymbol{x}}\} + \boldsymbol{K}_p\{\boldsymbol{x}_r - \boldsymbol{x}\} - \dot{\boldsymbol{J}}(\boldsymbol{\theta})\dot{\boldsymbol{\theta}}\right] \\ +\hat{\boldsymbol{H}}(\boldsymbol{\theta}, \dot{\boldsymbol{\theta}}) + \hat{\boldsymbol{G}}(\boldsymbol{\theta}) \end{aligned} \tag{2}$$

where, $\hat{}$ denotes the modeled term. $\boldsymbol{x} \in \Re^{6\times 1}$, $\dot{\boldsymbol{x}} \in \Re^{6\times 1}$ and $\ddot{\boldsymbol{x}} \in \Re^{6\times 1}$ are the position/orientation, velocity and acceleration vectors in Cartesian coordinate system, respectively. $\boldsymbol{x}_r$, $\dot{\boldsymbol{x}}_r$ and $\ddot{\boldsymbol{x}}_r$ are the desired position/orientation, velocity and acceleration vectors, respectively. $\boldsymbol{K}_v = \mathrm{diag}(K_{v1}, \ldots, K_{v6})$ and $\boldsymbol{K}_p = \mathrm{diag}(K_{p1}, \ldots, K_{p6})$ are the feedback gains of velocity and position, respectively. $\boldsymbol{J}(\boldsymbol{\theta})$ is the Jacobian which gives the relation $\dot{\boldsymbol{x}} = \boldsymbol{J}(\boldsymbol{\theta})\dot{\boldsymbol{\theta}}$. Note that $\boldsymbol{\theta}$, $\dot{\boldsymbol{\theta}}$, $\boldsymbol{x}$ and $\dot{\boldsymbol{x}}$ in (2) are actual values, i.e., controlled variables. The nonlinear compensation terms $\hat{\boldsymbol{H}}(\boldsymbol{\theta}, \dot{\boldsymbol{\theta}})$ and $\hat{\boldsymbol{G}}(\boldsymbol{\theta})$ are calculated to cancel the nonlinearity and are effective to achieve a stable trajectory control. Figure (3) shows the block diagram of the resolved acceleration control method, in which $Fkine(\boldsymbol{\theta})$ is the function to obtain the forward kinematics.

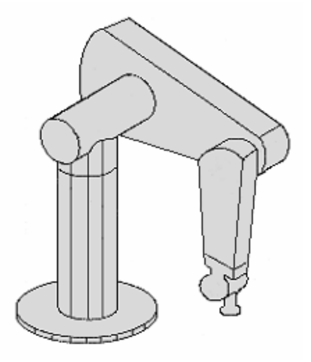

Fig. (2). PUMA560 manipulator.

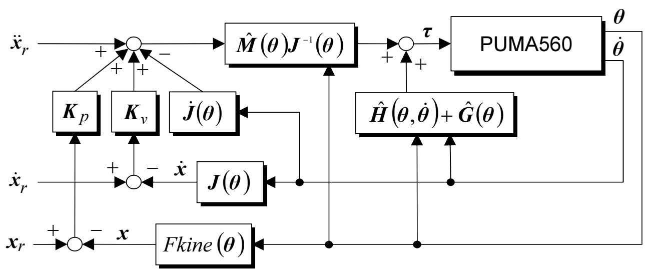

Fig. (3). Block diagram of the resolved acceleration control method, where $\boldsymbol{x}_r$, $\dot{\boldsymbol{x}}_r$ and $\ddot{\boldsymbol{x}}_r$ are the desired position/orientation, velocity and acceleration vectors in Cartesian coordinate system.

Fine Gain Tuning Considering the Critically Damped Condition

A basic gain tuning method considering the critically damped condition is explained giving an example. If it is assumed that the modeled terms ˆ are exact, then the nonlinearity of the robot dynamics can be canceled. Substituting (2) into (1) gives the following second order error system.

$$\ddot{\boldsymbol{e}} + \boldsymbol{K}_v\dot{\boldsymbol{e}} + \boldsymbol{K}_p\boldsymbol{e} = 0 \tag{3}$$

where, $\boldsymbol{e} = \boldsymbol{x}_r - \boldsymbol{x}$. Accordingly, $\boldsymbol{K}_v$ and $\boldsymbol{K}_p$ are selected suitably, e.g., considering the critically dumped condition given by

$$K_{vi} = 2\sqrt{K_{pi}}\ (i = 1, \dots, 6) \tag{4}$$

so that desirable responses can be obtained if under the ideal condition as given by (3). It should be noted, however, that there exists nonlinearity such as Coulomb friction and viscous friction. The robotic dynamic model with the friction term is written by

$$\boldsymbol{M}(\boldsymbol{\theta})\ddot{\boldsymbol{\theta}} + \boldsymbol{H}(\boldsymbol{\theta}, \dot{\boldsymbol{\theta}}) + \boldsymbol{G}(\boldsymbol{\theta}) + \boldsymbol{F}_r(\boldsymbol{\theta}, \dot{\boldsymbol{\theta}}) = \boldsymbol{\tau} \tag{5}$$

where, $\boldsymbol{F}_r(\boldsymbol{\theta}, \dot{\boldsymbol{\theta}}) \in \Re^{6\times 1}$ is the friction term composed of viscous friction and Coulomb friction. Also, because $\hat{\boldsymbol{M}}(\boldsymbol{\theta})$, $\hat{\boldsymbol{H}}(\boldsymbol{\theta}, \dot{\boldsymbol{\theta}})$ and $\hat{\boldsymbol{G}}(\boldsymbol{\theta})$ in (2) implicitly include undesirable modeled treatment error, the velocity and position gains in the servo portion should be more finely tuned after basic gain tuning. In the reminder of this section, genetic algorithms are applied to search for more suitable gains.

Desired Trajectory

For instance, in order to apply the resolved acceleration control method to the manipulator, the desired trajectory composed of $\boldsymbol{x}_r, \dot{\boldsymbol{x}}_r$ and $\ddot{\boldsymbol{x}}_r$ in Cartesian coordinate system must be prepared. First of all, the desired trajectory in Cartesian coordinate system is designed as shown in Fig. (4), in which the manipulator moves from the initial pose to the final pose. In this case, the robot stretches the arm tip 500 mm to the x-direction. The manipulator is still in initial and final poses. The desired trajectory $\boldsymbol{x}_r, \dot{\boldsymbol{x}}_r$ and $\ddot{\boldsymbol{x}}_r$ are calculated through a 4-1-4 order polynomial equation. We can design various trajectories with combination of acceleration time, constant velocity time and deceleration time. Desired angle $\boldsymbol{\theta}_r$, angle velocity $\dot{\boldsymbol{\theta}}_r$ and acceleration $\ddot{\boldsymbol{\theta}}_r$ in joint space are transformed from $\boldsymbol{x}_r, \dot{\boldsymbol{x}}_r$ and $\ddot{\boldsymbol{x}}_r$ by respectively using the inverse kinematics, Jacobian $\boldsymbol{J}(\theta)$ and sampling width Δt.

An example that consists of desired joint angle, angle velocity and acceleration is shown in Fig. (5), which are transformed from the $\boldsymbol{x}_r, \dot{\boldsymbol{x}}_r$ and $\ddot{\boldsymbol{x}}_r$ in Fig. (4). The trajectory in joint space makes the robot follow the motion as shown in Fig. (4). To obtain satisfactory and safe control performance without falling a singularity, $\boldsymbol{K}_v$ and $\boldsymbol{K}_p$ are roughly tuned in advance with trial and error, considering the combination around critically damped condition. We call this the initial manual tuning process. Two search spaces for $\boldsymbol{K}_v$ and $\boldsymbol{K}_p$ are obtained after the manual tuning process, so that $\boldsymbol{K}_v$ and $\boldsymbol{K}_p$ must be further tuned finely within the searched spaces to achieve a desirable motion without large overshoots and oscillations. In the next subsection, we propose a systematic tuning method using genetic algorithms which can be applied to after the manual tuning process.

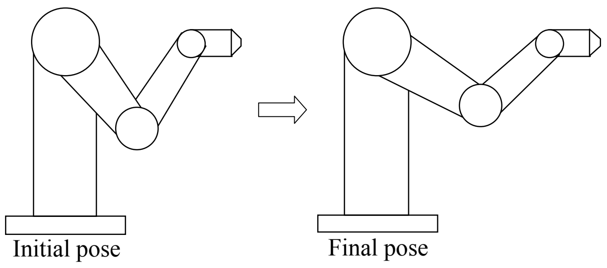

Fig. (4). Trajectory following control problem.

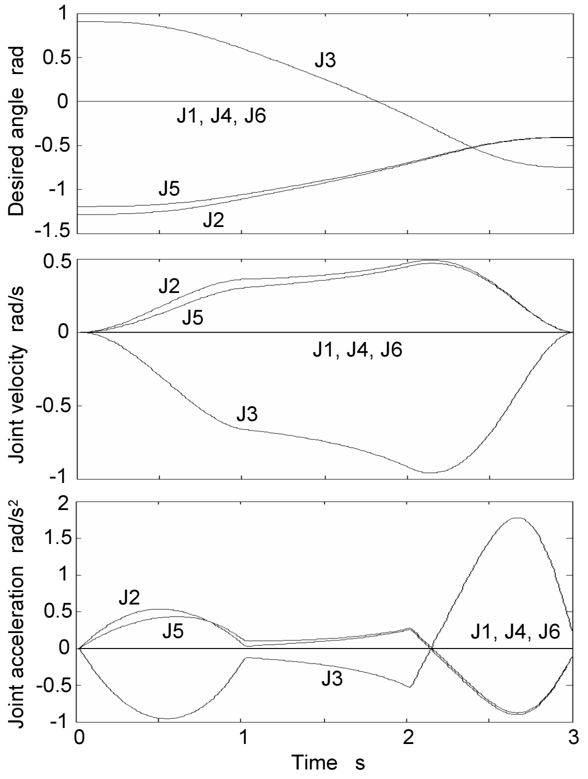

Fig. (5). Desired joint angle $\boldsymbol{\theta}_r$, velocity $\dot{\boldsymbol{\theta}}_r$ and acceleration $\ddot{\boldsymbol{\theta}}_r$ in joint space.

Fine Gain Tuning by Using Genetic Algorithms

The gain tuning after designing a robotic servo controller is an important and indispensable task to realize a smooth motion control system. The proposed gain tuning process consists of two steps. The first step is conducted to narrow down the search space for gains by operator's manual gain tuning while considering the circumference of the critically damped conditions given by (4). The ranges for gain

K_{v1}	K_{v2}	K_{v3}	K_{v4}	K_{v5}	K_{v6}	K_{p1}	K_{p2}	K_{p3}	K_{p4}	K_{p5}	K_{p6}

Fig. (6). Genotype of $\boldsymbol{K}_v$ and $\boldsymbol{K}_p$ which means an individual.

search are narrowed down between the minimum K_{vi}, K_{pi} and maximum K_{vi}, K_{pi}. The objective of the second step is to finely search for a more superior combination of K_{vi} and K_{pi} within the search space by using genetic algorithms [6], [7].

The genetic algorithm is used to optimize K_{vi} and $K_{pi} (i = 1, \dots, 6)$ which are the diagonal elements of $\boldsymbol{K}_v$ and $\boldsymbol{K}_p$. In order to apply genetic algorithms, K_{vi} and K_{pi} are transformed into a genotype. Diagonal elements K_{vi} and K_{pi} are encoded into 8 bits binary code and an individual is composed of $96 (= 8 \times 12)$ bits. A population has 60 individuals. Each individual in initial population is generated within expectable range considered in the initial manual tuning process. For example, we have preliminarily confirmed by randomly conducted manual tuning that if each element of $\boldsymbol{K}_v$ and $\boldsymbol{K}_p$ in (2) is set to $50 \leq K_{vi} \leq 200$ and $10 \leq K_{pi} \leq 50000$ $(i = 1, \dots, 6)$ then an undesirable singularity does not tend to occur. The singularity is a representative result after the system becomes unstable.

The elite survivable strategy is adopted in the selection process, where superior six individuals can survive to next generation as elites. Crossover and mutation which are the representative GA operations are not applied to the genotypes of the elites. Other remained 54 individuals are yielded by tournament selection. The genotype including the $\boldsymbol{K}_v$ and $\boldsymbol{K}_p$ is shown in Fig. (6). Each individual is evaluated by applying (2) to the trajectory following control problem shown in Fig. (4). The evaluation value E_v is calculated based on the square root of the sum of each directional error squared every sampling time, which is obtained by

$$E_v = \sum_{k=1}^{\frac{T}{\Delta t}} \sqrt{\sum_{i=1}^{3} \{e_i(k)\}^2} \tag{6}$$

where the error $e_i(k)$ $(i = 1, 2, 3)$ is the x-, y- and z-directional position errors in Cartesian space. T and Δt are the simulation time needed for a trajectory following control and sampling period, respectively. This is a minimization problem, where individuals with smaller fitness E_v can survive as elites to the next generation.

Tuning Result

Simulations were carried out by using the dynamic model of PUMA560 manipulator on MATLAB system. The robotic dynamics with the friction torque term given by (5) is applied. The friction torque term $\boldsymbol{F}_r$ consists of the viscous friction torque and Coulomb friction torque, which is represented by

$$\boldsymbol{F}_r(\boldsymbol{\theta}, \dot{\boldsymbol{\theta}}) = \boldsymbol{B}\boldsymbol{G}_r^2\dot{\boldsymbol{\theta}} + \boldsymbol{G}_r\boldsymbol{\tau}_c\{\mathrm{sign}(\dot{\boldsymbol{\theta}})\} \tag{7}$$

Table 1. Parameters of GA operation in case of resolved acceleration control law

Population size	60
Number of elites	6
Selection	Tournament selection
Crossover	Uniform crossover (rate=87.5%)
Mutation	Random mutation (rate=4.17%)
Search space of K_{vi}	$0.001 \leq K_{vi} \leq 220$
Search space of K_{pi}	$0.001 \leq K_{pi} \leq 40000$
Maximum generation	100

where $\boldsymbol{B}$ is the coefficient matrix of viscous friction at each motor, $\boldsymbol{G}_r$ is the reduction gear ratio matrix which represents the motor speed to joint speed, and $\boldsymbol{\tau}_c\{\mathrm{sign}(\dot{\boldsymbol{\theta}})\}$ is the Coulomb friction torque appeared at each motor. If $\mathrm{sign}(\dot{\theta}_i) > 0$, then $\tau_{ci} = \tau_{ci}^+$; if $\mathrm{sign}(\dot{\theta}_i) < 0$, then $\tau_{ci} = \tau_{ci}^-$. In simulations, $\boldsymbol{B}$ and $\boldsymbol{G}_r$ are set to diag(0.0015, 0.0008, 0.0014, 0.0001, 0.0001, 0.0000) and diag(-62.6, 107.8, -53.7, 76.0, 71.9, 76.7), respectively; $\boldsymbol{\tau}_c^+$ and $\boldsymbol{\tau}_c^-$ are set to $[0.395\ 0.126\ 0.132\ 0.011\ 0.009\ 0.004]^T$ and $[-0.435\ -0.071\ -0.105\ -0.017\ -0.015\ -0.011]^T$, respectively [3].

Before applying the GA operation, we broadly investigated the characteristics of the trajectory following control, while changing the combination of $\boldsymbol{K}_v$ and $\boldsymbol{K}_p$. As the result, the ranges in which the robot could move without falling a singularity were respectively $0.001 \leq K_{vi} \leq 220$ and $0.001 \leq K_{pi} \leq 40000$ $(i = 1, \dots, 6)$, so that the search space for GA was limited within these ranges. The uniform crossover was employed to the individuals with a high probability rate 87.5% (7/8) and the reverse action with mutation rate 4.17% (1/24) is given to all bits forming 54 individuals. Other parameters for GA operations are tabulated in Table 1. As an example of the tuning effectiveness, the variation of minimum fitness is shown in Fig. (7). It is observed that more excellent individuals have been selected through the alternation of generations with GA operations. After 100 generations, the phenotype of the best individual is represented by

$$\boldsymbol{K}_v = \mathrm{diag}(157, 204, 171, 181, 192, 76) \tag{8}$$

$$\boldsymbol{K}_p = \mathrm{diag}(23137, 7412, 24471, 7922, 13725, 5882) \tag{9}$$

When above gains were given, the minimum fitness E_v became 0.5241 by (6).

Figure (8) shows the simulation results which are performed by the gains with the minimum fitness. The upper figure in Fig. (8) shows the controlled angle trajectories of joints 2 and 3. The second figure similarly shows the angle velocities. The third and lower figures respectively illustrate the friction torques acting at the joints 2 and 3 which are given by (7). For example, it can be observed

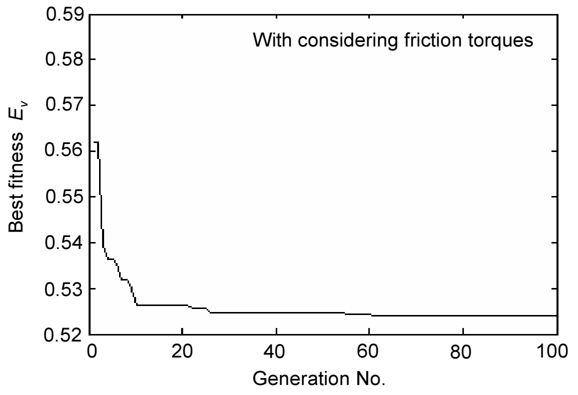

Fig. (7). Evolutionary history in case that the resolved acceleration control law was employed.

from the second figure in Fig. (8) that sign($\dot{\theta}_2$) and sign($\dot{\theta}_3$) are frequently changed around the start time and end time. According to the changes of sign($\dot{\theta}_2$) and sign($\dot{\theta}_3$), the large influence of Coulomb friction can be also observed around the corresponding time area in the third and lower figures. It is also observed that the best combination of K_{vi} and K_{pi} among obtained through the GA operation almost has the under damped condition given by

$$K_{vi} < 2\sqrt{K_{pi}} \ (i = 1, \ldots, 6) \tag{10}$$

This suggests that under damped situation is suitable for such a robotic trajectory following control with time-varying references as shown in Fig. (5).

In this section, we have preliminarily described the resolved acceleration controller as an example of robotic servo system which is surely incorporated in open architectural industrial robots. A simple but effective finely tuning method using genetic algorithms has been also introduced as one of intelligent control methodologies. Such a servo system is technically opened when an open architectural industrial robot is provided, so that we have only to make the desired trajectory $\boldsymbol{x}_r$, $\dot{\boldsymbol{x}}_r$ and $\ddot{\boldsymbol{x}}_r$ according to each task. Therefore, when a force control method is implemented, we have only to design the system so that the manipulated variables of the force control method are represented by $\boldsymbol{x}_r$, $\dot{\boldsymbol{x}}_r$ and $\ddot{\boldsymbol{x}}_r$ in discrete-time domain.

DYNAMIC SIMULATION

When a computer is used to control an industrial robot, the control law is generally represented by a discrete-time control system. In this section, it is described on how to simulate and evaluate the velocity-based discrete-time control system which is implemented into an open architectural industrial robot. Let's consider the impedance model following force control as an example of velocity-based discrete-time control systems. In order to conduct a simulation with a robotic dynamic model, manipulated variables written by velocity commands in discrete-time domain must be transformed into

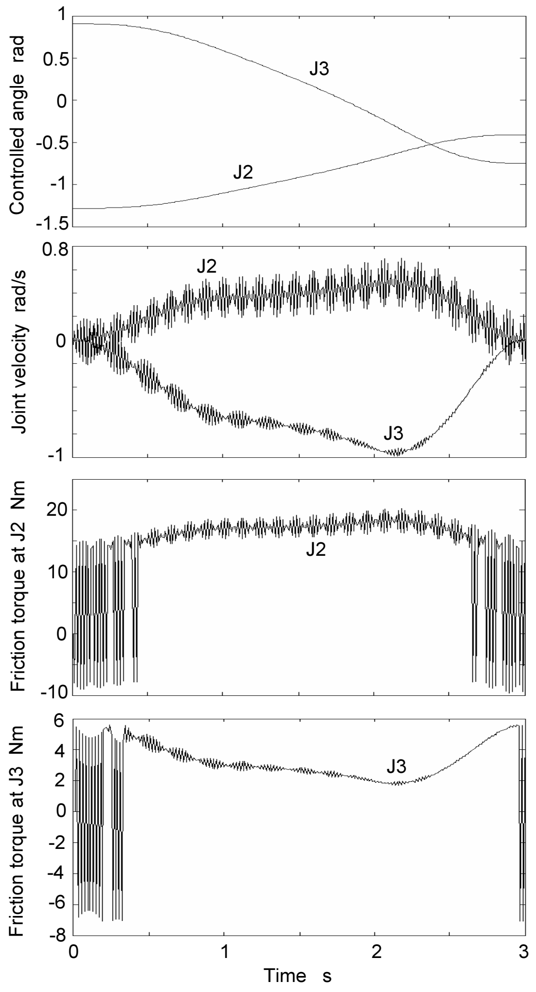

Fig. (8). Simulation result in case of using the gains with minimum fitness.

joint driving torques.

Impedance Model Following Force Control

Impedance control is one of the effective control strategies for a manipulator to desirably reduce or absorb the external force from an environment [8]. It is characterized by an ability which controls the mechanical impedance such as mass, damping and stiffness acting at joints. Impedance control does not have a force control mode or a position control mode but it is a combination of force and velocity. In order to control the contact force acting between the arm tip and environment, we have proposed the impedance model following force control methods (IMFFC) that can be easily implemented in industrial robots with an open architecture controller [9]. The desired impedance equation in Cartesian space for a robot manipulator is designed by

$$\begin{aligned}\boldsymbol{M}_d(\ddot{\boldsymbol{x}} - \ddot{\boldsymbol{x}}_d) + \boldsymbol{B}_d(\dot{\boldsymbol{x}} - \dot{\boldsymbol{x}}_d) + \boldsymbol{S}\boldsymbol{K}_d(\boldsymbol{x} - \boldsymbol{x}_d)\\ = \boldsymbol{S}\boldsymbol{F} + (\boldsymbol{I} - \boldsymbol{S})\boldsymbol{K}_f(\boldsymbol{F} - \boldsymbol{F}_d)\end{aligned} \tag{11}$$

where $\boldsymbol{x} \in \Re^3$, $\dot{\boldsymbol{x}} \in \Re^3$ and $\ddot{\boldsymbol{x}} \in \Re^3$ are the position, velocity and acceleration vectors, respectively. $\boldsymbol{M}_d \in \Re^{3\times3}$,

$\boldsymbol{B}_d \in \Re^{3\times 3}$ and $\boldsymbol{K}_d \in \Re^{3\times 3}$ are the coefficient matrices of desired mass, damping and stiffness, respectively. $\boldsymbol{F} \in \Re^3$ is the force vector. $\boldsymbol{K}_f \in \Re^{3\times 3}$ if the force feedback gain matrix. $\boldsymbol{x}_d$, $\dot{\boldsymbol{x}}_d$, $\ddot{\boldsymbol{x}}_d$ and $\boldsymbol{F}_d$ are the desired position, velocity, acceleration and force vectors, respectively. $\boldsymbol{S}$ is the switch matrix to select force control mode or compliance control mode. If $\boldsymbol{S} = \boldsymbol{0}$, (11) becomes force control mode in all directions; whereas if $\boldsymbol{S} = \boldsymbol{I}$ it becomes compliance control mode in all directions. Here, $\boldsymbol{I}$ is the identity matrix. $\boldsymbol{M}_d$, $\boldsymbol{B}_d$, $\boldsymbol{K}_d$ and $\boldsymbol{K}_f$ are set to positive-definite diagonal matrices.

When force control mode is selected in all directions, i.e., $\boldsymbol{S} = \boldsymbol{0}$, defining $\boldsymbol{X} = \dot{\boldsymbol{x}} - \dot{\boldsymbol{x}}_d$ gives

$$\dot{\boldsymbol{X}} = -\boldsymbol{M}_d^{-1}\boldsymbol{B}_d\boldsymbol{X} + \boldsymbol{M}_d^{-1}\boldsymbol{K}_f(\boldsymbol{F} - \boldsymbol{F}_d) \tag{12}$$

Here, the stability of (12) is briefly considered at the equilibrium by using Lyapunov stability analysis. The candidate of Lyapunov function [10] is proposed by

$$V(\boldsymbol{X}) = \frac{1}{2}\boldsymbol{X}^T\boldsymbol{X} \tag{13}$$

which is continuous and everywhere nonnegative. Differentiating (13) gives

$$\begin{aligned}\dot{V}(\boldsymbol{X}) = \boldsymbol{X}^T\dot{\boldsymbol{X}} &= \boldsymbol{X}^T(-\boldsymbol{M}_d^{-1}\boldsymbol{B}_d\boldsymbol{X})\\ &= -\boldsymbol{X}^T\boldsymbol{M}_d^{-1}\boldsymbol{B}_d\boldsymbol{X}\end{aligned} \tag{14}$$

which is everywhere non-positive since $\boldsymbol{M}_d^{-1}\boldsymbol{B}_d$ is a positive definite diagonal matrix. Therefore, (13) is indeed a Lyapunov function for the system given by (12). $\dot{V}(\boldsymbol{X})$ can be zero only at $\boldsymbol{X} = \boldsymbol{0}$, everywhere else $V(\boldsymbol{X})$ decreases, so that (12) is asymptotically stable.

In general, (12) can be resolved as

$$\begin{aligned}\boldsymbol{X} &= e^{-\boldsymbol{M}_d^{-1}\boldsymbol{B}_d t}\boldsymbol{X}(0)\\ &+ \int_0^t e^{-\boldsymbol{M}_d^{-1}\boldsymbol{B}_d(t-\tau)}\boldsymbol{M}_d^{-1}\boldsymbol{K}_f(\boldsymbol{F} - \boldsymbol{F}_d)d\tau\end{aligned} \tag{15}$$

In the following, we consider the form in the discrete time k using a sampling time Δt. If it is assumed that $\boldsymbol{M}_d$, $\boldsymbol{B}_d$, $\boldsymbol{K}_f$, $\boldsymbol{F}$ and $\boldsymbol{F}_d$ are constant at $\Delta t(k-1) \le t < \Delta tk$, then defining $\boldsymbol{X}(k) = \boldsymbol{X}(t)|_{t=\Delta tk}$ leads to the recursive equation given by

$$\begin{aligned}\boldsymbol{X}(k) &= e^{-\boldsymbol{M}_d^{-1}\boldsymbol{B}_d\Delta t}\ \boldsymbol{X}(k-1)\\ &- \left(e^{-\boldsymbol{M}_d^{-1}\boldsymbol{B}_d\Delta t} - \boldsymbol{I}\right)\boldsymbol{B}_d^{-1}\boldsymbol{K}_f\{\boldsymbol{F}(k) - \boldsymbol{F}_d\}\end{aligned} \tag{16}$$

Remembering $\boldsymbol{X} = \dot{\boldsymbol{x}} - \dot{\boldsymbol{x}}_d$, giving $\dot{\boldsymbol{x}}_d = \boldsymbol{0}$ in the direction of force control, and adding an integral action, the equation of velocity command in terms of Cartesian space is derived by

$$\begin{aligned}\dot{\boldsymbol{x}}(k) &= e^{-\boldsymbol{M}_d^{-1}\boldsymbol{B}_d\Delta t}\ \dot{\boldsymbol{x}}(k-1)\\ &- \left(e^{-\boldsymbol{M}_d^{-1}\boldsymbol{B}_d\Delta t} - \boldsymbol{I}\right)\boldsymbol{B}_d^{-1}\boldsymbol{K}_f\{\boldsymbol{F}(k) - \boldsymbol{F}_d\}\\ &+\boldsymbol{K}_i\sum_{n=1}^{k}\{\boldsymbol{F}(n) - \boldsymbol{F}_d\}\end{aligned} \tag{17}$$

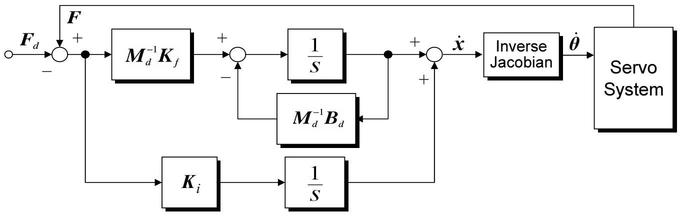

Fig. (9). Block diagram of impedance model following force control with I-action.

where $\boldsymbol{K}_i \in \Re^{3\times 3}$ is the integral gain matrix and is also set to a positive-definite diagonal matrix. The impedance model following force control method written by (17) is used to control the force which an industrial robot gives an environment. The block diagram of (17) is shown in Fig. (9). As can be seen, the force is regulated by a feedback control loop.

How to Generate the References for Servo System

To simulate the IMFFC given in discrete-time domain, the velocity $\dot{\boldsymbol{x}}(k)$ calculated from (17) must be transformed into joint driving torques. Accordingly, we propose a transformation technique where $\dot{\boldsymbol{x}}(k)$ is given to the reference value of the Cartesian-based servo controller. The joint driving torques transformed from $\dot{\boldsymbol{x}}(k)$ control each joint independently. Figure (10) shows the block diagram of the transformation technique. First of all, the reference position $\boldsymbol{x}_r(k)$ is obtained from

$$\boldsymbol{x}_r(k) = \dot{\boldsymbol{x}}(k) + \boldsymbol{x}(k) \tag{18}$$

where $\boldsymbol{x}(k)$ is the current position of the arm tip. Then, the velocity $\dot{\boldsymbol{x}}_r(k)$ and acceleration $\ddot{\boldsymbol{x}}_r(k)$ are made by using a discrete time k and sampling width Δt.

$$\dot{\boldsymbol{x}}_r(k) = \{\boldsymbol{x}_r(k) - \boldsymbol{x}_r(k-1)\}/\Delta t \tag{19}$$

$$\ddot{\boldsymbol{x}}_r(k) = \{\dot{\boldsymbol{x}}_r(k) - \dot{\boldsymbol{x}}_r(k-1)\}/\Delta t \tag{20}$$

For example, it is assumed that the resolved acceleration control law is used in the servo system of an industrial robot. In this case, $\boldsymbol{x}_r(k)$, $\dot{\boldsymbol{x}}_r(k)$ and $\ddot{\boldsymbol{x}}_r(k)$ generated from Eqs. (18)∼(20) are respectively given to the references of the servo system as shown in Fig. (10), so that the joint torque $\boldsymbol{\tau}$ is generated from the resolved acceleration control law given by (2). Thus, the proposed technique allows us to implement the IMFFC in the computer simulations. The interpretation of Fig. (10) is as follows: The IMFFC makes a velocity $\dot{\boldsymbol{x}}(k)$ using the force error $\boldsymbol{F}(k) - \boldsymbol{F}_d$. Also, current position $\boldsymbol{x}(k)$ is obtained from the forward kinematics of PUMA560 manipulator. $\dot{\boldsymbol{x}}(k)$ and $\boldsymbol{x}(k)$ are summed up, and which is given to the desired position $\boldsymbol{x}_r(k)$ of the resolved acceleration controller used as a servo system. As mentioned above, the desired velocity and acceleration respectively given to $\dot{\boldsymbol{x}}_r(k)$ and $\ddot{\boldsymbol{x}}_r(k)$ are calculated by using $\boldsymbol{x}_r(k)$ and Δt.

Manipulated variables written by velocity commands in discrete-time domain can be transformed to joint driving

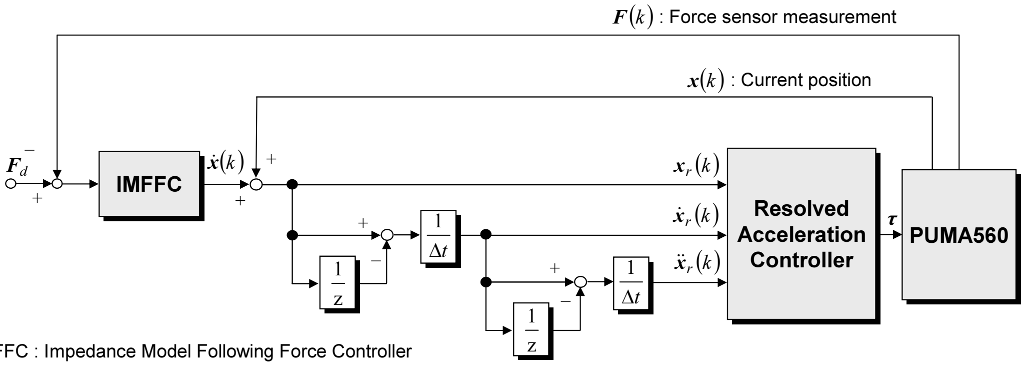
$F(k)$: Force sensor measurement
$x(k)$: Current position
F_d
IMFFC
$\dot{x}(k)$
$x_r(k)$
$\dot{x}_r(k)$
$\ddot{x}_r(k)$
$\frac{1}{\Delta t}$
$\frac{1}{z}$
Resolved Acceleration Controller
τ
PUMA560
IMFFC : Impedance Model Following Force Controller

Fig. (10). Block diagram of impedance model following force controller, in which $\boldsymbol{x}_r(k)$, $\dot{\boldsymbol{x}}_r(k)$ and $\ddot{\boldsymbol{x}}_r(k)$ are transformed into the joint driving torque $\boldsymbol{\tau}$.

torques due to the proposed scheme as shown in Fig. (10), so that various velocity-based control methodologies written in discrete-time domain developed for open architectural industrial robots have been able to be easily simulated.

IN CASE OF FUZZY CONTROL

Fuzzy Force Control

To improve the force control performance in the environment with unknown dynamics or curved surface, how to construct the fuzzy controller by using the servo system introduced in the previous section is first described. The main feature of the fuzzy force controller is that it generates proper position compensations as feedforward command so that overshoots and oscillations in the direction of force control can be suppressed satisfactorily. A profiling control simulation using the dynamics of PUMA560 manipulator is shown to demonstrates the effectiveness under an environment with time-varying stiffness.

If a force controlled manipulator is applied to a profiling task, the change of environmental stiffness or shape causes undesirable force error. The fuzzy force controller works to reduce the force error. The fuzzy controller generates velocity vector, i.e., position compensation vector $\Delta\boldsymbol{x}(k) = [\Delta x(k)\ \Delta y(k)\ \Delta z(k)]^T$ as a feedforward control according to the force error and its rate every sampling time. For example, let's consider the x-directional compensation. Fuzzy inputs are the force error and its rate defined as

$$e_x(k) = F_{dx} - F_x(k) \tag{21}$$

$$\Delta e_x(k) = \frac{\{e_x(k) - e_x(k-1)\}}{\Delta t} \tag{22}$$

where F_{dx} and $F_x(k)$ are the desired force and sensed force, respectively. Following the fuzzy approach, if the information on only x-direction is used, the fuzzy rules are described by

Rule 1 IF $e_x(k)$ is $\tilde{A}_1$ and $\Delta e_x(k)$ is $\tilde{B}_1$, THEN $\Delta x(k) = c_1$
Rule 2 IF $e_x(k)$ is $\tilde{A}_2$ and $\Delta e_x(k)$ is $\tilde{B}_2$, THEN $\Delta x(k) = c_2$
Rule 3 IF $e_x(k)$ is $\tilde{A}_3$ and $\Delta e_x(k)$ is $\tilde{B}_3$, THEN $\Delta x(k) = c_3$

$\vdots$

Rule L IF $e_x(k)$ is $\tilde{A}_L$ and $\Delta e_x(k)$ is $\tilde{B}_L$, THEN $\Delta x(k) = c_L$

where $\tilde{A}_i(i = 1, ..., L)$ and $\tilde{B}_i$ are the i-th antecedent fuzzy sets for two fuzzy inputs $e_x(k)$ and $\Delta e_x(k)$; c_i is the consequent constant value at the i-th rule; L is the fuzzy rule number. The confidence of the antecedent part at the i-th rule is obtained by

$$\omega_i = \mu_{\tilde{A}i}\{e_x(k)\} \wedge \mu_{\tilde{B}i}\{\Delta e_x(k)\} \tag{23}$$

where $\mu_X(\bullet)$ denotes the confidence of a fuzzy set labeled by X. Therefore, the fuzzy position compensation is calculated by

$$\Delta x(k) = \frac{\sum_{i=1}^{L} \omega_i c_i}{\sum_{j=1}^{L} \omega_j} \tag{24}$$

The position compensation of other directions, $\Delta y(k)$ or $\Delta z(k)$ is similarly calculated by the same procedure. Note that, the fuzzy set used is the following Gaussian type membership function

$$\mu_X(x) = \exp\{\log(0.5)(x-\alpha)^2\beta^2\} \tag{25}$$

where α is the center of membership function and β is the reciprocal value of standard deviation. Figures (11) and (12) show the designed antecedent membership functions for $e_x(k)$ and $\Delta e_x(k)$, respectively. Each reciprocal value of the standard deviation are 66.7 and 100, respectively. The corresponding constant values in consequent part are tabulated in Table 2. It is expected that the proposed fuzzy force controller will improve the force control performance.

Simulation of Fuzzy Force Control

In this subsection, a profiling control as shown in Fig. (13) is conducted and analyzed in detail. And the promise of the proposed method shown in Fig. (15) is evaluated through the profiling control simulation.There are two important factors that prevent the stable profiling control. The one is that

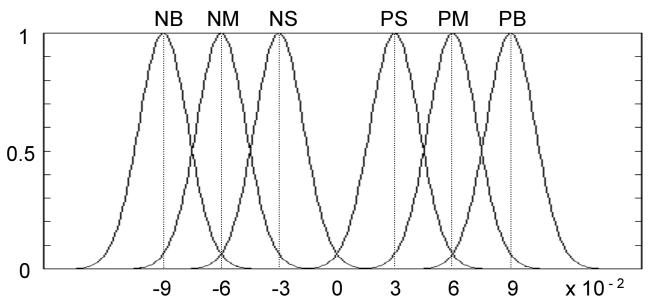

Fig. (11). Antecedent membership functions for $e_x(k)$.

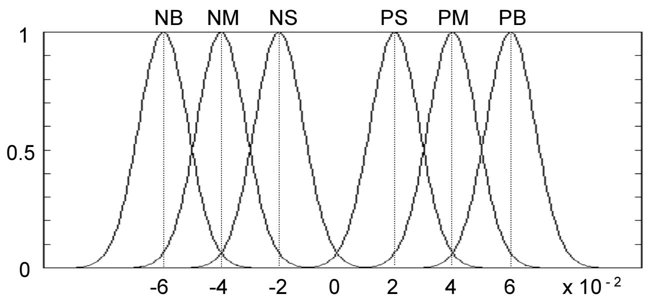

Fig. (12). Antecedent membership functions for $\Delta e_x(k)$.

the dynamics of the object is almost unknown, and perhaps changes frequently. The other is that how curved surface the object has. In practice, robots for polishing and sanding must deal with such objects. It is assumed that the object with an inclined and flat plane is fixed in the robot workspace as shown in Fig. (13). The end-effector contacts with the object; it is tried to control the contact force from normal direction of the slope to converge to a reference of 1 N. It is also assumed that the contact force $\boldsymbol{F}(k)$ is generated by

$$\boldsymbol{F}(k) = -\boldsymbol{B}_m \dot{\boldsymbol{x}}(k) - \boldsymbol{K}_m\{\boldsymbol{x}(k) - \boldsymbol{x}_m\} \tag{26}$$

where $\boldsymbol{B}_m$ Ns/m and $\boldsymbol{K}_m$ N/m are the viscosity and stiffness coefficients of the object to be positive definite diagonal matrices, and $\boldsymbol{x}_m$ is the initial contact position. In the following simulation, the stiffness of the object $\boldsymbol{K}_m = \mathrm{diag}(\mathrm{K_{mx}},\ \mathrm{K_{my}},\ \mathrm{K_{mz}})$ varies as follows:

$$K_{mx} \;=\; 100000 + 30000 \times \sin(3k\pi) \tag{27}$$

Table 2. Constant values in consequent part [$\times 10^{-4}$ mm]

$e_x(k)$ \ $\Delta e_x(k)$	NB	NM	NS	PS	PM	PB
NB	-16.0	-14.0	-12.0	-8.0	-6.0	-4.0
NM	-9.6	-8.4	-7.2	-4.8	-3.6	-2.4
NS	-3.2	-2.8	-2.4	-1.6	-1.2	-0.8
PS	0.8	1.2	1.6	2.4	2.8	3.2
PM	2.4	3.6	4.8	7.2	8.4	9.6
PB	4.0	6.0	8.0	12.0	4.0	16.0

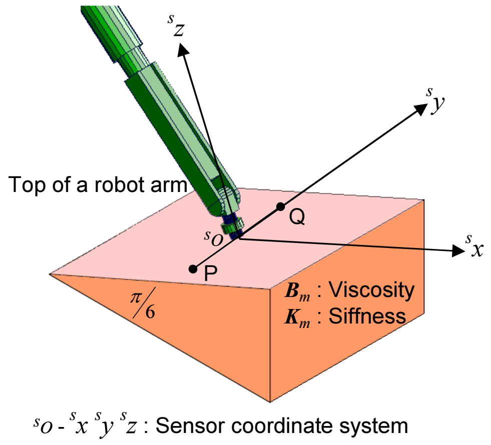
^sz
^sy
^sx
^so
Top of a robot arm
Q
P
$\pi/6$
$\boldsymbol{B}_m$: Viscosity
$\boldsymbol{K}_m$: Siffness
^so - ^sx ^sy ^sz : Sensor coordinate system

Fig. (13). Profiling control situation.

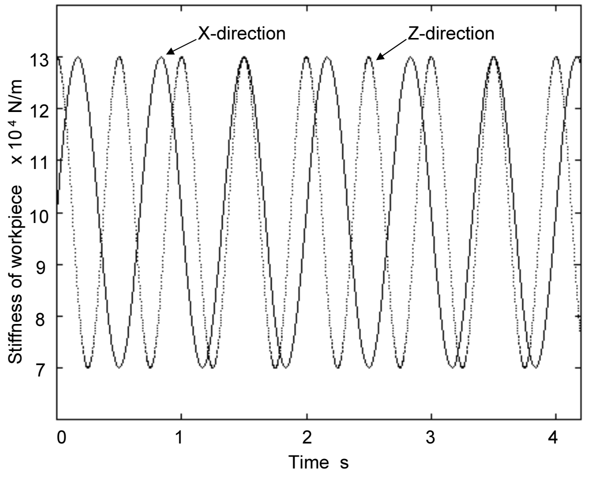

Fig. (14). Time histories of environmental stiffness in x- and z-directions.

$$K_{mz} \;=\; 100000 + 30000 \times \sin(4k\pi + \pi/2) \tag{28}$$

which is illustrated as shown in Fig. (14). The viscosity is fixed to $\boldsymbol{B}_m = \mathrm{diag}(15, 15, 15)$ Ns/m. Here, we control the z-directional contact force and y-directional trajectory (i.e., from point P to Q shown in Fig. (13)) in sensor coordinate system. Also, we set up the manipulator as PUMA560 [3], its dynamics is given by

$$\boldsymbol{M}_x(\boldsymbol{\theta})\ddot{\boldsymbol{x}} + \boldsymbol{H}_x(\boldsymbol{\theta}, \dot{\boldsymbol{\theta}}) + \boldsymbol{G}_x(\boldsymbol{\theta}) = \boldsymbol{J}^{-T}(\boldsymbol{\theta})\boldsymbol{\tau} + \boldsymbol{F} \tag{29}$$

where $\boldsymbol{H}_x(\boldsymbol{\theta})$ is the Coriolis and centrifugal forces, $\boldsymbol{G}_x(\boldsymbol{\theta})$ is the gravity term, $\boldsymbol{J}(\boldsymbol{\theta})$ is the Jacobian matrix, and $\boldsymbol{\tau}$ is the joint driving torque. Note that the subscript x means the value in Cartesian space. The dynamics is solved by the Runge-Kutta method on Matlab system. In this case, the resolved acceleration controller is used in the servo system shown in Fig. (15), and the $\boldsymbol{x}_r(k)$ is generated by

$$\boldsymbol{x}_r(k) = \dot{\boldsymbol{x}}(k) + \Delta\boldsymbol{x}(k) + \boldsymbol{x}(k) \tag{30}$$

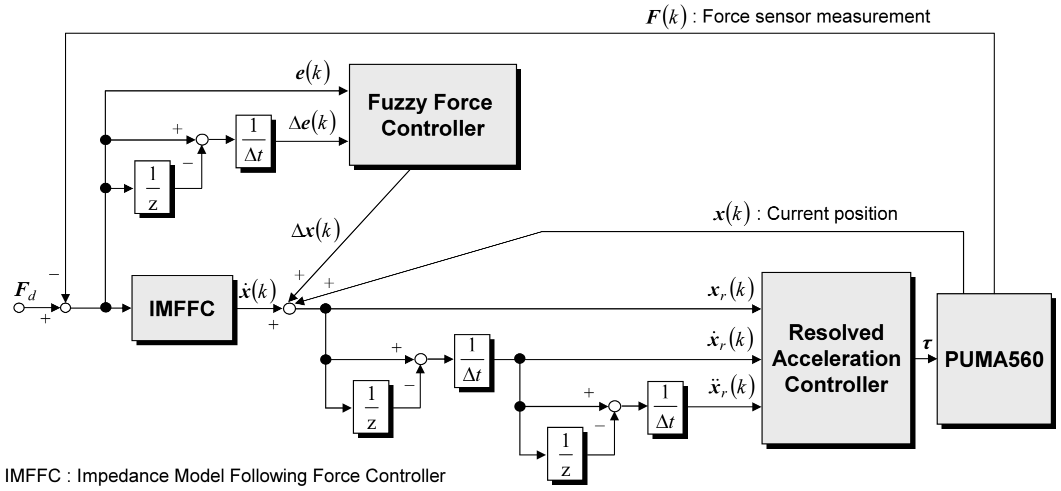
$F(k)$: Force sensor measurement
$e(k)$
$\Delta e(k)$
Fuzzy Force Controller
$\Delta x(k)$
$x(k)$: Current position
F_d
IMFFC
$\dot{x}(k)$
$x_r(k)$
$\dot{x}_r(k)$
$\ddot{x}_r(k)$
Resolved Acceleration Controller
τ
PUMA560
IMFFC : Impedance Model Following Force Controller

Fig. (15). Block diagram of the force control system using a fuzzy reasoning.

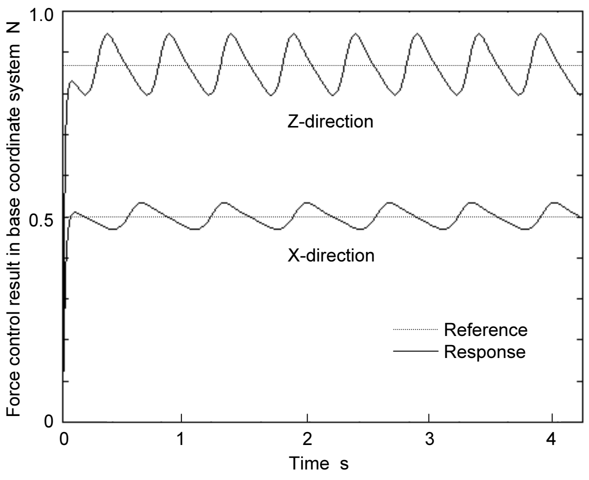

Fig. (16). Force control result using only IMFFC.

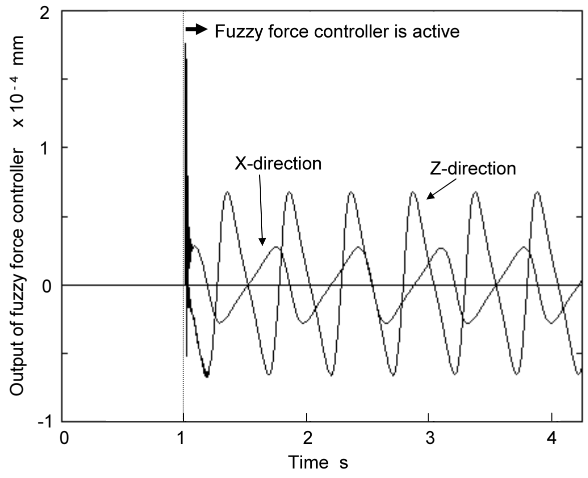

Fig. (18). Position compensations $\Delta x(k)$ and $\Delta z(k)$ generated from the fuzzy force controller.

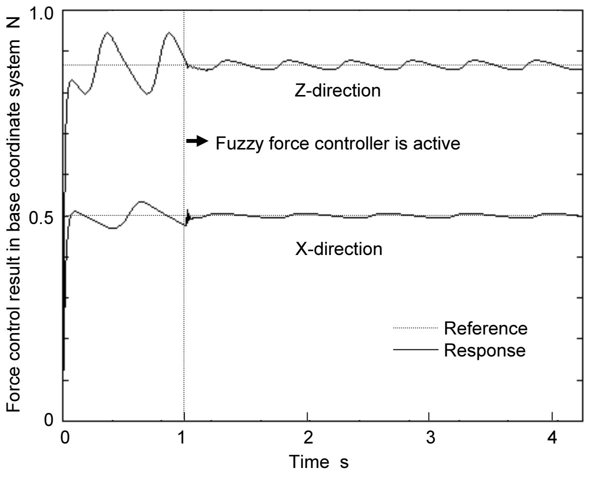

Fig. (17). Force control result using IMFFC with fuzzy force controller.

As a first case study, the fuzzy force controller was not used. In this case, it was difficult to suppress the oscillations caused by the change of the object stiffness. Figure (16) shows the force control result. It should be noted that the z-directional contact force in sensor coordinate system is represented by x- and z-directional forces in robot base coordinate system. Another case study was carried out to evaluate the ability of the fuzzy force controller. The fuzzy force controller was set to active after the passage of one second. The force control result is shown in Fig. (17). In this case, x- and z-directional position compensations $\Delta x(k)$ and $\Delta z(k)$ generated from the fuzzy force controller are shown in Fig. (18). It is observed that each directional force acting on the end -effector is desirably improved in spite of the variation of the object stiffness.

Table 3. Simulation condition for acquiring teaching signals

Approaching velocity	18.4 mm/s
Desired contact force F_d	20 N
Stiffness of object	100, 500, 1000 N/m,
Desired damping of IMFFC B_d	Trial and error

IN CASE OF NEURAL NETWORK

Neural Network Force Control

Next, a force control system using a neural network (NN) is presented to realize a desirable contact motion with high speed response characteristics between a manipulator and an environment. For example, the PC-based controller used in the robot sander provides API (Application Programming Interface) functions to control the position/orientation of the arm tip with velocity commands [11]. Making good use of the API functions, it is expected that if the control method is described by a velocity-based command, then the method can be easily applied to actual open architectural industrial robots. Thus, if the neural network considered in this section is made up of a model whose output is the quantity of velocity, it will be used for the actual robot with the open architectural controller without difficulty.

First of all, in order to acquire teaching signals, we consider a contact control problem as shown in Fig. (13), in which the tip of the PUMA560 comes in contact with the environment from normal direction with a low speed and then it is stabilized with a desired contact force by using the IMFFC. The desired teaching signals are ideal responses without overshoots and oscillations, which are suitable for training patterns of the neural network. Simulations on contact motion were repeatedly conducted with trial and error under the condition tabulated in Table 3. Through the simulation with an environmental stiffness K_m=1000 N/m, a desirable response of force error ${}^s e_{fz}(k) = F_{dz} - F_z(k)$ as shown in Fig. (19) was obtained along normal direction. Note that ^{s} denotes the sensor coordinate system and ${}^s\boldsymbol{e}_f(k)$ is composed of $[{}^s e_{fx}(k)\ {}^s e_{fy}(k)\ {}^s e_{fz}(k)]^T$. In this case, the z-directional velocity ${}^s\dot{z}(k)$ yielding at the arm tip was shown in Fig. (20). The results given by Figs. (19) and (20) are one pattern of teaching signals for input and output, respectively. Other two teaching signals for inputs are $\Delta^s e_{fz} = {}^s e_{fz}(k) - {}^s e_{fz}(k-1)$ and ${}^s\dot{z}(k-1)$, respectively. Similarly, other teaching signals with the condition of K_m=500 and 100 N/m were respectively obtained through simulations.

As an example, we design a neural network model with three inputs and one output in z-direction of sensor coordinate system. The three inputs are the force error ${}^s e_{fz}(k)$, its increment $\Delta^s e_{fz}(k)$ and velocity of arm tip ${}^s\dot{z}(k-1)$ in z-direction of sensor coordinate system as illustrated in Fig. (13). Also, the output of NN is the velocity ${}^s\dot{z}_{nn}(k)$. In order to learn the ideal contact motion, a four-layered recurrent neural network composed of an input layer, two hidden layers and an output layer is used as shown in Fig. (21). The two hidden layers have twenty units respectively. The input to the hidden layers or the output layer is a weighted sum of the previous layer. The sum is squashed into ±0.5 by the following nonlinear activation function:

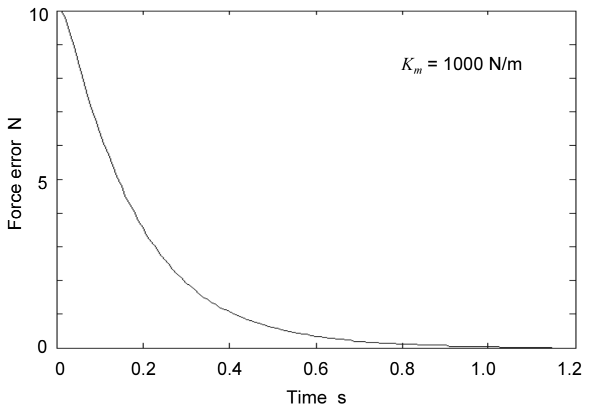

Fig. (19). A desirable force response ${}^s E_{fz}(k)$ for a teaching signal obtained through simulation.

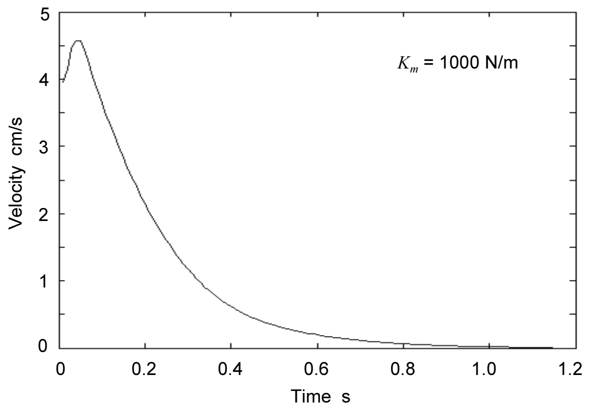

Fig. (20). Velocity of arm tip ${}^s\dot{z}(k)$, which is the teaching signal of output of neural network.

$$f(X) = \frac{1}{1+\exp(-X)} - 0.5 \tag{31}$$

The input/output relation of each unit is given by

$$X_{i,l} = \sum_{j=1}^{n} w_{j,l-1}^{i,l} o_{j,l-1} \tag{32}$$

where $X_{i,l}$ is the state of ith unit in lth layer. $w_{j,l-1}^{i,l}$ is the interconnection weight between the ith unit in lth layer and the jth unit in $(l-1)$th layer. $o_{j,l-1}$ is the output of the jth unit in $(l-1)$th layer. n is the number of units in $(l-1)$th layer. The neural network was trained by the back propagation algorithm to learn the mapping between force and velocity responses without overshoot and oscillation. The

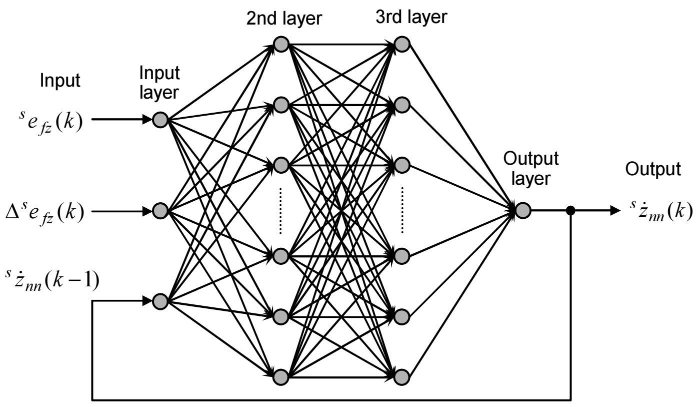

Fig. (21). Neural network for dynamics learning of contact motion.

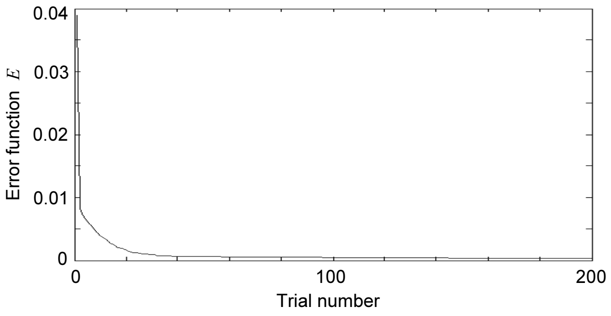

Fig. (22). Learning history of error function E.

learning was iterated until error function E became small sufficiently. Figure (22) shows the learning history, in which the error function E is calculated by

$$E = \sum_{k=1}^{N} \frac{|^s\dot{z}(k) - {}^s\dot{z}_{nn}(k)|}{N} \tag{33}$$

where N is the number of the training patter, i.e., total discrete time in the simulation. After the learning process, the NN could give an output as shown in Fig. (23) similar to the teaching signal.

Simulation of Neural Network Force Controller

Figure (25) shows the block diagram of the force control system using the learned neural network, in which the output from the neural network is feedforwardly added to the output of the IMFFC. A force control result in case of K_m=1000 N/m is shown in Fig. (24). It is observed that the response is improved and the raising time is shortened by using the learned neural network. In this case, IMFFC+NN, NN only and IMFFC only generate manipulated variables as shown in Fig. (26), rspectively. The neural network feedforwardly generates desired manipulated variables. However, if the environment had a stiffer dynamics than the learned environment, then the contact characteristics tended to become worse. In such a case, an adaptability against to unlearned environment should be considered. As an example,

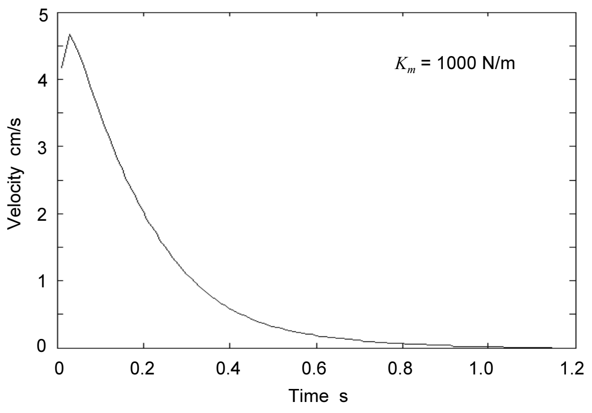

Fig. (23). Output $^s\dot{z}_{nn}(k)$ generated from the learned neural network.

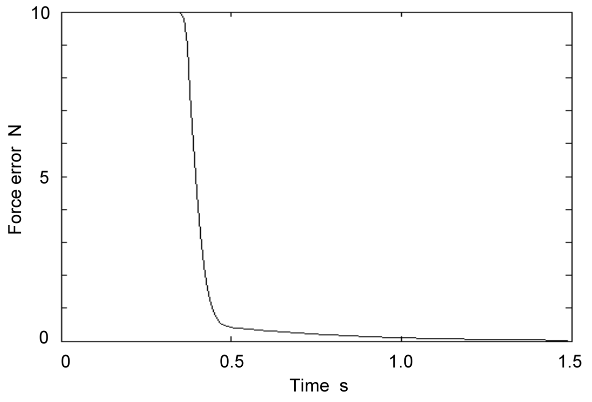

Fig. (24). An example of force control result by feedforwardly using the learned NN.

the feedback error learning of neural network will be effective to carry out an on-line learning [12], [13].

CONCLUSIONS

In this article, a simulation technique of velocity-based discrete-time control system for open architectural industrial robots has been presented by giving and combining examples of intelligent controls such as genetic algorithms, fuzzy control and neural network. In order to develop a novel control system for an open architectural industrial robot, it is required from the points of view concerning safety, cost and easiness to preliminarily examine and evaluate the characteristics and performance. In such a case, the proposed simulation technique will be useful.

It is important and required to realize a simple and accurate simulation environment for industrial robots with an open architectural controller. When a computer is used to control an industrial robot, the control law is generally represented by a discrete-time control system. For this reason, it has been considered on how to simulate and evaluate the discrete-time control system which will be implemented into an open architectural industrial robot. In order to carry out a simulation with a robotic dynamic model, we have

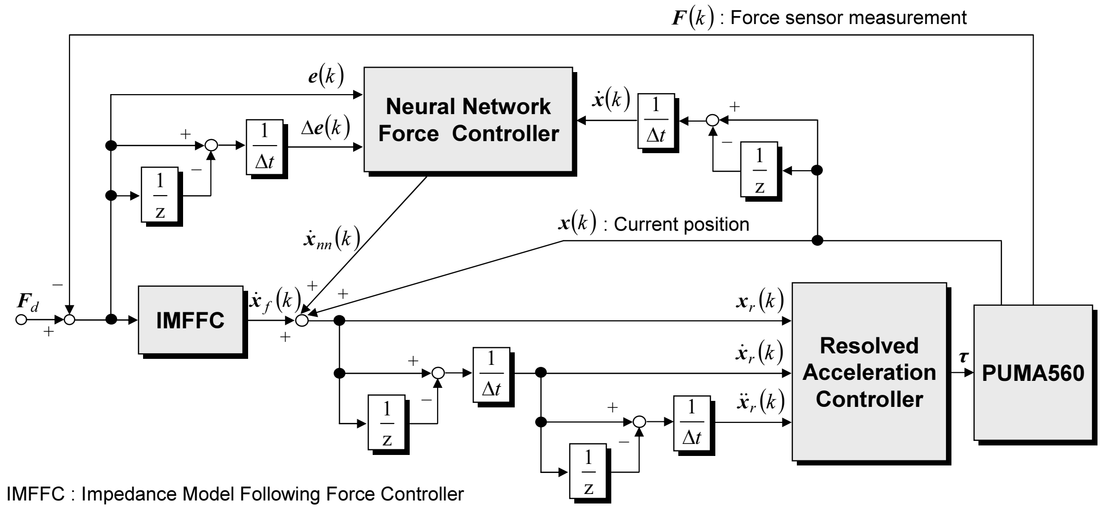
$F(k)$: Force sensor measurement
$e(k)$
Neural Network Force Controller
$\dot{x}(k)$
$\Delta e(k)$
$\frac{1}{\Delta t}$
$\frac{1}{z}$
$x(k)$: Current position
$\dot{x}_{nn}(k)$
F_d
IMFFC
$\dot{x}_f(k)$
$x_r(k)$
$\dot{x}_r(k)$
$\ddot{x}_r(k)$
Resolved Acceleration Controller
τ
PUMA560
IMFFC : Impedance Model Following Force Controller

Fig. (25). Block diagram of the force control system using a neural network.

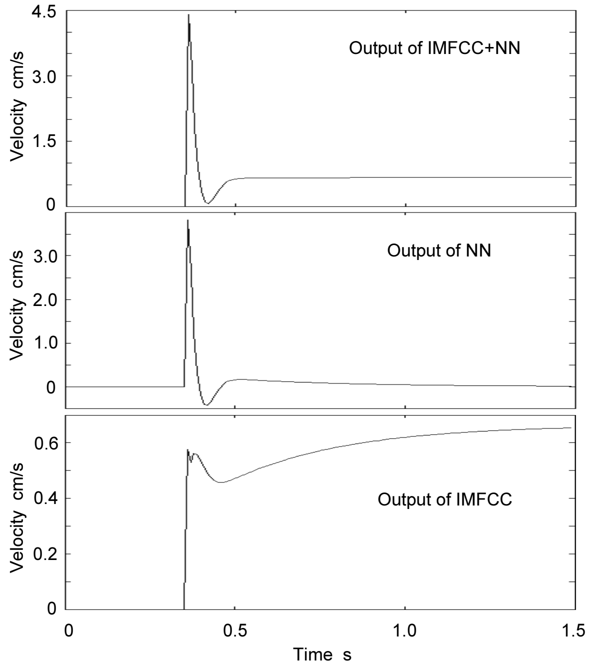

Fig. (26). The upper, middle and lower figures show the outputs from IMFFC+NN, NN only and IMFFC only, respectively.

shown the scheme in detail which transforms the velocity-based manipulated variables into joint driving torques. The effectiveness and promise have been also demonstrated by simulations using a dynamic model of the PUMA560 manipulator. Preliminary evaluation of various new control approaches will be realized easily and safely for industrial robots with an open architectural controller due to the proposed simulation technique.

REFERENCES

[1] C. Chen, M.M. Trivedi, C.R. Bidlack, "Simulation and animation of sensor-driven robots," *IEEE Trans. on Robotics and Automation*, vol. 10, no. 5, pp. 684–704, 1994.

[2] F. Benimeli, V. Mata, F. Valero, "A comparison between direct and indirect dynamic parameter identification methods in industrial robots," *Robotica*, vol. 24, no. 5, pp. 579–590, 2006.

[3] P. Corke, "A Robotics Toolbox for MATLAB," *IEEE Robotics & Automation Magazine*, vol. 3, no. 1, pp. 24–32, 1996.

[4] P. Corke, "MATLAB Toolboxes: robotics and vision for students and teachers," *IEEE Robotics & Automation Magazine*, vol. 14, no. 4, pp. 16–17, 2007.

[5] J.Y.S. Luh, M.H. Walker, R.P.C. Paul, "Resolved acceleration control of mechanical Manipulator," *IEEE Trans. on Automatic Control*, vol. 25, no. 3, pp. 468–474, 1980.

[6] F. Nagata, K. Kuribayashi, K. Kiguchi and K. Watanabe, "Simulation of fine gain tuning using genetic algorithms for model-based robotic servo controllers," *Procs. of the IEEE Int. Symp. on Computational Intelligence in Robotics and Automation*, pp. 196–201, 2007.

[7] F. Nagata, I. Okabayashi, M. Matsuno, et al., "Fine gain tuning for model-based robotic servo controllers using genetic algorithms," *Procs. of the 13th Int. Conf. on Advanced Robotics*, pp. 987–992, 2007.

[8] N. Hogan, "Impedance control: An approach to manipulation: Part I - Part III," *Trans. of the ASME, Journal of Dynamic Systems, Measurement and Control*, vol. 107, pp. 1–24, 1985.

[9] F. Nagata, K. Watanabe, S. Hashino, et al., "Polishing robot using a joystick controlled teaching system," *Procs. of the IEEE Int. Conf. on Industrial Electronics, Control and Instrumentation*, pp. 632–637, 2000.

[10] J.J. Craig, Introduction to ROBOTICS —Mechanics and Control Second Edition—, Reading MA: Addison Wesley Publishing Co., 1989.

[11] F. Nagata, Y. Kusumoto, Y. Fujimoto and K. Watanabe, "Robotic sanding system for new designed furniture with free-formed surface," *Robotics and Computer-Integrated Manufacturing*, vol. 23, no. 4, pp. 371–379, 2007.

[12] M. Kawato, "The feedback-error-learning neural network for supervised motor learning," Advanced Neural Computers, Elsevier Amsterdam, pp. 365–373, 1990.

[13] J. Nakanishi, S. Schaal, "Feedback error learning and nonlinear adaptive control," *Neural Networks*, vol. 17, no. 10, pp. 1453–1465, 2004.

Received: June 12, 2008 Revised: June 30, 2008 Accepted: July 10, 2008