Suboptimal Control of Industrial Manipulators with a Weighted Minimum Time-Fuel Criterion

BYUNG KOOK KIM, MEMBER, IEEE, AND KANG G. SHIN, SENIOR MEMBER, IEEE

Abstract—**Even if a manipulator does not have to follow a prespecified path (i.e., a time history of position and velocity) due to the complexity and nonlinearity of the manipulator dynamics, control of manipulators has been conventionally divided into two subproblems, namely *path planning* and *path tracking*, which are then *separately* and *independently* solved. This may result in mathematically tractable solutions but cannot offer a solution that utilizes manipulators' maximum capabilities (e.g., operating them at their maximum speed).**

To combat this problem, we have developed a suboptimal method for controlling manipulators that provides improved performance in both their operating speed and use of energy. The nonlinearity and the joint couplings in the manipulator dynamics—a major hurdle in the design of robot control—are handled by a new concept of averaging the dynamics at each sampling interval. With the averaged dynamics, we have derived a feedback controller which has a simple structure allowing for on-line implementation with inexpensive mini- or microcomputers, and offers a near minimum time-fuel (NMTF) solution, thus enabling manipulators to perform nearly up to their maximum capability and efficiency.

As a demonstrative example, we have simulated the proposed control method with a dynamic model of the Unimation PUMA 600 series manipulator on a DEC VAX-11/780. The simulation results agree with the expected high performance nature of the control method.

I. INTRODUCTION

RECENTLY, robotics has emerged as an important field in engineering mainly because of its high potential for improving both the goals of manufacturing productivity and working environment. Industrial manipulators are computer-controlled mechanical devices and are the primary component in contemporary automation systems. It is therefore essential to design optimal manipulator systems with a suitable performance criterion which is consistent with the foregoing goals.

The performance of manipulators can be bettered by improving their mechanical construction and/or by using more effective controllers. In this paper we are only concerned with the latter. Although manipulator control problems can in general be classified into four different types depending upon if 1) they have to follow a prespecified path and/or 2) they operate in a collision-free workspace (see [10] for details), there are many applications which do not require robotic manipulators to strictly follow a prescribed path and, also, collision with obstacles can be avoided by specifying a few appropriate intermediate points in the workspace for the manipulator to pass through [11]. Consequently, in such a case the manipulator control problem can be converted to a more general form in which the manipulator is given freedom to move along any path between any two given intermediate or end points.

In view of the preceding fact, for many cases manipulators are desired to move from one point to another as fast and with as little energy as possible. Consequently, it is important to design an efficient controller which requires less time and energy, thus pushing the manipulators to be operated at their maximum efficiency, i.e., with maximum operating speed and minimum energy. This consideration naturally leads to an optimal control problem of robotic manipulators with a minimum time-fuel criterion. However, it is in general very difficult to obtain an exact closed-form optimal solution to the problem since the dynamics of manipulators are highly nonlinear coupled functions of their positions and velocities, and also of their payloads. There are two alternative approaches conceivable for this problem:

1) off-line minimum time path planning followed by on-line path tracking, [1] and

2) derivation of a suboptimal controller with realistic approximations of the manipulator dynamics.

For the first approach, when an optimal path planner is available, one can use one of many well-known, on-line path tracking algorithms [4], [5], [9], which drive the manipulator to follow the prescribed path with the prescheduled velocities. These path tracking algorithms are based on the computed torque technique using Lagrangian or Newton–Euler formulation of manipulator dynamics. Also, there are a few known optimal path planners which determine a time history of desired joint position and velocity by minimizing the total traveling time for a given sequence of specified positions (describing the desired path) in joint coordinates [2], or in Cartesian coordinates [3], with *global bounds* on velocity and acceleration. Unlike path tracking algorithms, at the time of this writing there are no known methods for path planning which include the manipulator dynamics. Note that the maximum speed and acceleration of a manipulator vary with its position, payload, and configuration. For example, an optimal path for a manipulator has to be generated on the basis of the maximum speed allowed under the worst (global) condition, since it may otherwise not be able to follow the prespecified path with the prescheduled velocities. This implies that path planning has to be made with the *global least upper bounds* of all possible manipulator's speeds and accelerations. Therefore, the full capability of the manipulator cannot be utilized if this approach is taken.

The second approach can be adopted to nearly fully utilize the capability of individual manipulators. Only a few attempts have been made in this context due mainly to the difficulty in obtaining amenable solutions. Kahn and Roth [1] linearized the manipulator dynamics at the final target point, and used the decoupled dynamic model to derive a near minimum-time controller. This method suffers from the fact that the linearized dynamic equations would

Manuscript received August 24, 1983; revised January 16, 1984. This paper is based on a prior submission of March 3, 1983. Paper recommended by J. Y. Luh, Past Associate Editor for Automation and Robotics. This work was supported in part by the U.S. Air Force Office of Scientific Research under Contract F49620-82-C-0089, the Postdoctoral Program of the Korea Science and Engineering Foundation, Republic of Korea, and the Robot Systems Division, Center for Robotics and Integrated Manufacturing (CRIM), The University of Michigan, Ann Arbor, MI.

B.K. Kim is with Woojn Instrument Corporation, Seoul, Korea.

K. G. Shin is with the Department of Electrical Engineering and Computer Science, The University of Michigan, Ann Arbor, MI 48109.

[1] As one reviewer pointed out, this alternative is converting the general endpoint control problem to a restricted path-following problem. This is mainly because the latter allows for easy handling of complex dynamics, e.g., off-line computation of nominal input torques or linearization of the dynamic equations with respect to the specified path.

0018-9286/85/0100-0001$01.00

not be valid if the manipulator is not located in the vicinity of the final target. Consequently, this method is acceptable only when manipulator motion is confined to a small region in the neighborhood of the target point. Lynch [6] developed a minimum-time controller for sequential axis operation. Only one axis was moved at one time, simplifying the dynamic equations of the manipulator (e.g., linear time-invariant equations for a two-axis cylindrical manipulator), and hence the controller. However, the sequential control requires much more time (approximately n times more for an n-joint manipulator) than the simultaneous control of all joints. Also some results were derived on the basis of minimum energy control with a fixed terminal time [7], [8], in which they used oversimplified dynamic models without considering manipulator's operating speed.

For the second approach to manipulator control one may also attempt to use the methods known for handling the nonlinearity of aircraft engine control; that is sequential linearizations of the nonlinear dynamics near a set of steady-state operating conditions for perturbation equations as well as application of gain scheduling method as evidenced in [17], [18]. However, this design method is not directly applicable to the manipulator control problem in which extremal values of the control magnitude (such as bang-bang control) are required.

The second approach with a suitable suboptimal controller, if possible, is highly desirable due to the possible full-utilization of manipulator's capability. In this paper, we have adopted the second approach and have developed a suboptimal *feedback controller* for industrial manipulators with the weighted minimum time-fuel (MTF) criterion. The choice of this criterion is justified by its direct link to the goal of improving productivity and saving energy. The near-minimum time-fuel (NMTF) controller is derived in *feedback form* with a judicious approximation in the calculation of switching curves for the controller. (Ideally, the switching curves have to be derived from the overall, exact manipulator dynamics.) Although the optimal controller is developed for continuous time domain, the manipulator dynamics are updated at each of discrete sampling instants. The above approximation of the dynamics is motivated by the fact that almost all manipulator control algorithms are nowadays implemented, hopefully in real-time, on digital computers, but the required computation for their exact realizations is prohibitively complex and time-consuming. Consequently, in order to cope with the nonlinearity and joint couplings in the manipulator dynamics, the model parameters of the manipulator are updated at each sampling interval on the basis of feedback information of positions and velocities. Then, an *averaged dynamics* concept—which utilizes all available dynamic information of the current and the final states to update the dynamics continually at each sampling interval—is newly introduced to design the proposed NMTF controller. Since the switching curves are derived for the current interval and then updated at the next sampling interval with feedback (of position and velocity), 1) the approximation error at a sampling instant is implicitly compensated, and 2) the averaging process can effectively handle the nonlinearity and joint couplings in the manipulator dynamics. As will be seen later, the update of the manipulator dynamics at each sampling interval and the averaged dynamics are both simple, and therefore regarded suitable for real-time implementation.

The main contribution of this paper lies in that 1) the complexity of the manipulator dynamics—which has been a major obstacle in the design of robot control—is handled effectively by regular updates of the dynamics with the averaging process; 2) its structural simplicity allows for on-line implementation with mini- and microcomputers; 3) it has high potential for improving the manipulator efficiency in both operating speed and use of energy; and 4) it results in a closed-loop feedback controller, whereas most existing ones are open-loop MTF controllers.

This paper is organized as follows. In Section II the algorithm for suboptimal control of manipulators with the MTF criterion is derived, and the method of updating the dynamic parameters with the averaged dynamics concept is presented. Also considered is the synchronization of each joint controller for simultaneous convergence of all manipulator joints to a target point. In Section III we analyze the amount of computation required for implementing the NMTF algorithm, particularly exploring the possibility of real-time implementation. Section IV presents the simulation of the NMTF controller with a dynamic model of the Unimation PUMA 600 series manipulator, and the paper concludes with Section V.

II. The Near—Minimum Time-Fuel Controller

A. Derivation of the Controller

Using the Lagrangian mechanics, one can derive explicit manipulator dynamic equations which relate generalized forces/torques to the joint positions, velocities, and accelerations

$$\boldsymbol{D}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{h}(\boldsymbol{q}, \dot{\boldsymbol{q}}) + \boldsymbol{g}(\boldsymbol{q}) = \boldsymbol{u} \tag{1}$$

where $\boldsymbol{u}$ is an $n \times 1$ generalized force/torque vector, $\boldsymbol{q}$, $\dot{\boldsymbol{q}}$, $\ddot{\boldsymbol{q}}$ are $n \times 1$ vectors of generalized coordinates, velocities, and accelerations, respectively. $\boldsymbol{D}(\boldsymbol{q})$ is an $n \times n$ inertia matrix, $\boldsymbol{h}(\boldsymbol{q}, \dot{\boldsymbol{q}})$ is an $n \times 1$ Coriolis and centrifugal force vector, $\boldsymbol{g}(\boldsymbol{q})$ is an $n \times 1$ gravitational loading vector, and n is the number of joints in the manipulator. The inertia, the gravity loading, and the Coriolis and centrifugal terms depend on the position of each joint as well as on the mass, first moment, and inertia of each link. These terms are also functions of manipulator's payload (i.e., tool and parts). The dynamic equations (1) can be converted to a state-variable representation with a $2n$-dimensional state vector $\boldsymbol{y} = [(\boldsymbol{y}^p)^T(\boldsymbol{y}^v)^T]^T = [\boldsymbol{q}^T\dot{\boldsymbol{q}}^T]^T$ (T denotes here transpose) as follows:

$$\dot{\boldsymbol{y}} = \boldsymbol{a}(\boldsymbol{y}) + \boldsymbol{B}(\boldsymbol{y})\boldsymbol{u} \tag{2}$$

where

$$\boldsymbol{a}(\boldsymbol{y}) = \begin{bmatrix} \boldsymbol{y}^p \\ -\boldsymbol{D}^{-1}(\boldsymbol{y}^p)[\boldsymbol{g}(\boldsymbol{y}^p) + \boldsymbol{h}(\boldsymbol{y}^p, \boldsymbol{y}^v)] \end{bmatrix}$$

$$\boldsymbol{B}(\boldsymbol{y}) = \begin{bmatrix} 0 \\ \boldsymbol{D}^{-1}(\boldsymbol{y}^p) \end{bmatrix}.$$

Each joint of the manipulator is separately driven by an electric motor or by a hydraulic motor, and naturally there exist certain limits in the magnitudes of driving forces or torques. Hence, constraints on the input vector $\boldsymbol{u}$ can be represented in general

$$\boldsymbol{u}^- \le \boldsymbol{u} \le \boldsymbol{u}^+$$

where $\boldsymbol{u}^-$ and $\boldsymbol{u}^+$ are $n \times 1$ vectors representing the minimum and the maximum values of input force/torque, respectively. This vector inequality denotes n inequalities component-wise, i.e., $u_j^- \le u_j \le u_j^+$, $j = 1, 2, \cdots, n$. Then, the weighted time-fuel optimal control problem can be stated as follows.

Problem 1: Find control $\boldsymbol{u}^*(t)$, $t_0 \le t \le t_f$, such that the system given by (2) is steered to a given target point

$$\boldsymbol{y}(t_f) = \boldsymbol{y}_f \tag{4}$$

from a given initial point.

$$\boldsymbol{y}(t_0) = \boldsymbol{y}_0 \tag{5}$$

while minimizing the performance index

$$J(\boldsymbol{u}) = \int_{t_0}^{t_f} \left[\lambda + \sum_{j=1}^{n} |u_j| \right] dt \tag{6}$$

subject to the input constraint (3).

Note that this is an open terminal-time problem (i.e., t_f is left

free), and the parameter λ is introduced to set a relative weight between time and fuel. When the value of λ approaches zero, this becomes the fuel-optimal control problem, and when the value of λ becomes infinity, this approaches the time-optimal control problem.

Using Pontryagin's maximum principle [12], we can derive necessary conditions for the optimal solution as follows. By minimizing the Hamiltonian functional for Problem 1,

$$H(y(t), u(t), p(t), t) = \lambda + \sum_{j=1}^{n} |u_j(t)| + p^T(t)[a(y(t)) + B(y(t))u(t)] \tag{7}$$

we can obtain the optimal control $u^*(t)$ satisfying the following inequality for all $t \in [t_0, t_f]$:

$$|u_j^*(t)| + p^{*T}(t)b_j(y^*(t))u_j^*(t) \leq |u_j(t)| + p^{*T}b_j(y^*(t))u_j(t), \qquad \text{for } j = 1, 2, \cdots, n \tag{8}$$

where $y^*(t)$ is the optimal state at time t, $b_j(y^*(t))$ the jth column of $B(y^*(t))$, and $p^*(t)$ is the costate vector determined by

$$\dot{p}^*(t) = -\frac{\partial H}{\partial y^*}. \tag{9}$$

The properties used in this derivation are that 1) the components of u are mutually independent of one another (this is true in the manipulator control); 2) the Hamiltonian functional is linear in control; and 3) control is bounded. Therefore, the form of optimal control is for $j = 1, 2, \cdots, n$

$$u_j^*(t) = \begin{cases} u_j^- & \text{for } p^{*T}(t)b_j(y^*(t)) > 1 \\ \text{nonpositive} & \text{for } p^{*T}(t)b_j(y^*(t)) = 1 \\ 0 & \text{for } -1 < p^{*T}(t)b_j(y^*(t)) < 1 \\ \text{nonnegative} & \text{for } p^{*T}(t)b_j(y^*(t)) = -1 \\ u_j^+ & \text{for } p^{*T}(t)b_j(y^*(t)) < -1. \end{cases} \tag{10}$$

Substituting u_j of (10) into (2), we can obtain a $4n$-dimensional differential equation with $y^*(t)$, $p^*(t)$ and the boundary conditions described by (4) and (5), resulting in a two-point boundary value problem; this is in general very difficult, if not impossible, to solve. Only numerical solutions for all but extremely simple cases may be obtained due to the nonlinearity and inertial couplings in the manipulator dynamics.

Due to this difficulty, direct application of the maximum principle is not pursued here. Instead, we consider first optimal control of an approximate dynamic system for each individual axis and then update the concerned dynamics with feedback information to compensate for the errors induced by the approximation as well as for the nonlinearity in the dynamics. The system dynamic model, i.e., (1) can be rewritten

$$\sum_{i=1}^{n} d_{ji}(q)\ddot{q}_i = u_j - f_j(q, \dot{q}), \qquad j = 1, 2, \cdots, n \tag{11}$$

where q_j is the jth element of q, $d_{ji}(q)$ is the (j, i)th element of $D(q)$, $f_j(q, \dot{q})$ is the jth element of $f(q, \dot{q}) = g(q) + h(q, \dot{q})$.

The joints are coupled in (11) by inertia terms, direct use of which complicates the control system design. Instead, we rearrange (11) into

$$\ddot{q}_j = \alpha_j(q)u_j + \beta_j(q, \dot{q}, u), \qquad j = 1, 2, \cdots, n \tag{12}$$

where $\alpha_j(q) = D_{jj}^{-1}(q)$, $\beta_j(q, \dot{q}, u) = \sum_{i=1, i\neq j}^{n} D_{ji}^{-1}(q)u_i - \sum_{i=1}^{n} D_{ji}^{-1}(q)f_i(q, \dot{q})$, and D_{ij}^{-1} denotes the (i, j)th element of D^{-1}. The second term of (12) represents the coupling effects from other joints on the jth joint as well as Coriolis, centrifugal, and gravitational forces, which are the major hindrance in obtaining amenable optimal solutions to the manipulator control problem. However, (12) can be regarded as an uncoupled subsystem model for the jth joint of the manipulator if the value of $\beta_j(q, \dot{q}, u)$ is calculated or approximated by some judicious means. In such a case Problem 1 can be solved, yielding a computationally simple solution that is implementable in real-time with mini- or microcomputers. To this end, we will in this paper pursue a method for calculating approximate values of both $\alpha_j(q)$, and $\beta_j(q, \dot{q}, u)$, which are nonlinear functions of the manipulator position, velocity, and input.

The optimal control problem for each joint system of the manipulator is then stated as follows.

Problem 2: Find the control $u_j^*(t)$, $0 \leq t \leq t_f$, such that the jth joint system with the state vector $(x_j^p, x_j^v)^T = (q_j, \dot{q}_j)^T$

$$\begin{aligned} \dot{x}_j^p &= x_j^v \\ \dot{x}_j^v &= \alpha_j(q)u_j + \beta_j(q, \dot{q}, u) \end{aligned} \tag{13}$$

is steered to a given target point $x_{jf} = (q_{jf}, \dot{q}_{jf})^T$ from an initial point $x_{j0} = (q_{j0}, \dot{q}_{j0})^T$, while minimizing the performance index $J_j(u_j) = \int_{t_0}^{t_f} [\lambda_j + |u_j|]dt$ subject to the input constraint $u_j^- \leq u_j \leq u_j^+$.

Since the values of $\alpha_j(q)$, and $\beta_j(q, \dot{q}, u)$ are nonlinear functions of the manipulator position, velocity, and input, it is still impossible to obtain a closed-form solution to Problem 2. However, it would be possible to obtain a closed-form optimal solution if the values of α_j and β_j are time-invariant. Also, in this case, we can synthesize the solution in feedback form which is essential in robotic applications to effectively handle the manipulator dynamics that vary widely with its position and payload. Consequently, we 1) assume both α_j and β_j to be constant over one sampling interval, [2] and 2) modify them at each update time to include the preceding nonlinear dependence on q, $\dot{q}$, and u. The latter is made on the basis of input as well as position and velocity feedbacks, and more on this will be discussed in Section II-B. Hence, Problem 2 will be solved for a constant, continuous-time system which is modified at each of discrete sampling intervals.

The optimal solution to Problem 2 when $\alpha_j(q)$ and $\beta_j(q, \dot{q}, u)$ are time-invariant, i.e., $\ddot{q}_j = \alpha_j u_j + \beta_j$, can be synthesized in feedback form as below. The solution is obtained by extending the known optimal solution of the minimum time-fuel control problem with dynamics of $\ddot{q} = u$ as in [12], but three cases should be considered separately according to the magnitudes of λ_j and β_j. Note that this system is controllable and the solution has no singularity region. The optimal solution is as follows.

A) When $\lambda_j > |\beta_j|$.

i) When $z_j^v \geq 0$

$$u_j^*(t) = \begin{cases} u_j^+ & \text{if } z_j^p \leq \delta_j^-(z_j^v)^2/2\gamma_j^- \\ 0 & \text{if } (z_j^v)^2/2\gamma_j^- > z_j^p > \delta_j^-(z_j^v)^2/2\gamma_j^- \\ u_j^- & \text{if } z_j^p \geq (z_j^v)^2/2\gamma_j^-. \end{cases} \tag{14}$$

ii) When $z_j^v < 0$

$$u_j^*(t) = \begin{cases} u_j^+ & \text{if } z_j^p \leq (z_j^v)^2/2\gamma_j^+ \\ 0 & \text{if } (z_j^v)^2/2\gamma_j^+ < z_j^p < \delta_j^+(z_j^v)^2/2\gamma_j^+ \\ u_j^- & \text{if } z_j^p \geq \delta_j^+(z_j^v)^2/2\gamma_j^+ \end{cases} \tag{15}$$

where

[2] The parameters can be updated at every mth sampling interval if the update at each sampling interval is computationally too demanding. m has to be determined by both the real-time requirements and the given computation power.

$$z_j^p = x_j^p - x_{jf}^p$$
$$z_j^v = x_j^v - x_{jf}^v$$
$$\gamma_j^+ = \alpha_j u_j^+ + \beta_j$$
$$\gamma_j^- = \alpha_j u_j^- + \beta_j$$
$$\delta_j^+ = 1 + \frac{4\lambda_j \alpha_j u_j^+}{(\lambda_j + \beta_j)^2}$$
$$\delta_j^- = 1 - \frac{4\lambda_j \alpha_j u_j^-}{(\lambda_j - \beta_j)^2}.$$

These equations indicate that $u_j^*(t)$ $j = 1, 2, \cdots, n$ can be determined in feedback form with a set of switching curves as illustrated in Fig. 1.

B) When $\beta_j > \lambda_j > 0$.

i) When $z_j^v \geq 0$

$$u_j^*(t) = \begin{cases} 0 & \text{if } z_j^p < (z_j^v)^2/2\gamma_j^- \\ u_j^- & \text{if } z_j^p \geq (z_j^v)^2/2\gamma_j^-. \end{cases} \quad (16)$$

ii) When $z_j^v < 0$

$$u_j^*(t) = \begin{cases} u_j^+ & \text{if } \delta_j^-(z_j^v)^2 \leq z_j^p < (z_j^v)^2/2\beta_j \\ 0 & \text{if } z_j^p/2\gamma_j^- < \delta_j^-(z_j^v)^2/2\gamma_j^- \text{ or } z_j^p = (z_j^v)^2/2\beta_j \\ u_j^- & \text{if } z_j^p > (z_j^v)^2/2\beta_j. \end{cases} \quad (17)$$

These equations are illustrated in Fig. 2.

C) When $-\beta_j > \lambda_j > 0$.

i) When $z_j^v \geq 0$

$$u_j^*(t) = \begin{cases} u_j^+ & \text{if } z_j^p < (z_j^v)^2/2\beta_j \\ 0 & \text{if } z_j^p = (z_j^v)^2/2\beta_j \text{ or } z_j^p > \delta_j^+(z_j^v)^2/2\gamma_j^+ \\ u_j^- & \text{if } (z_j^v)^2/2\beta_j < z_j^p \leq \delta_j^+(z_j^v)^2/2\gamma_j^+. \end{cases} \quad (18)$$

ii) When $z_j^v < 0$

$$u_j^*(t) = \begin{cases} u_j^+ & \text{if } z_j^p \leq (z_j^v)^2/2\gamma_j^+ \\ 0 & \text{if } z^p > (z_j^v)^2/2\gamma_j^+. \end{cases} \quad (19)$$

These equations are illustrated in Fig. 3.

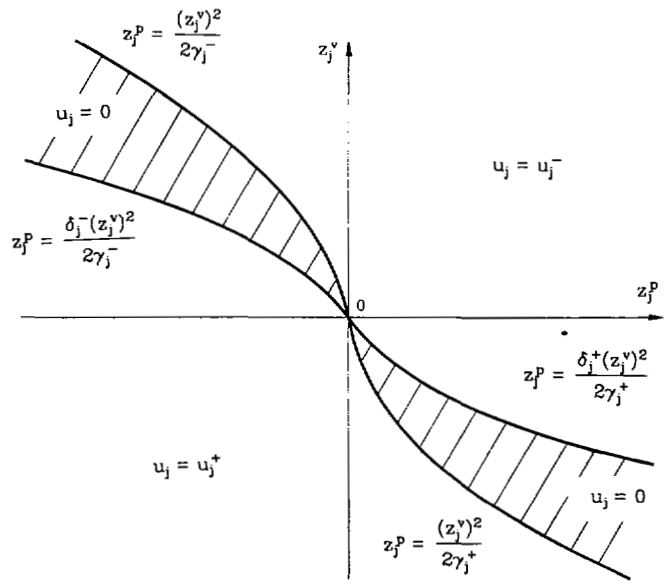


Fig. 1. Switching curves when $\lambda > |\beta|$.

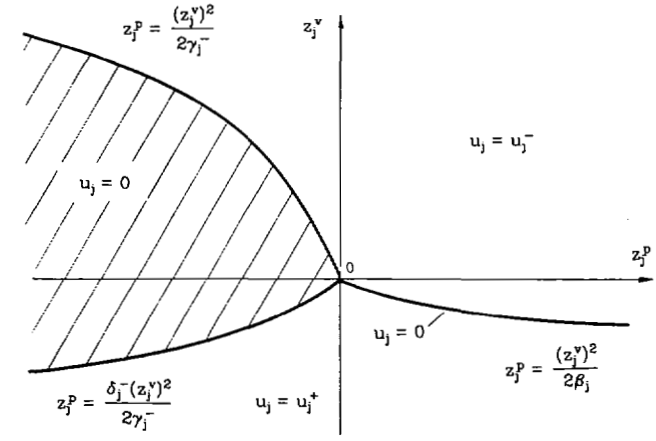


Fig. 2. Switching curves when $\beta > \lambda$.

B. The Averaged Dynamics

In order to utilize the above solution for the case when α_j and β_j are time-invariant, we have developed a method for 1) approximating their values at time t, $\bar{p}_j(t) = [\bar{\alpha}_j(t), \bar{\beta}_j(t)]^T$, that represent an average behavior of the manipulator dynamics from the current state to the final target state, and 2) continuously updating those values to cope with the variation in the manipulator dynamics due to their nonlinear dependence on q, $\dot{q}$, and u. We call this approximation the *averaged dynamics method,* which is described below. At time t_1, using the current position $y^p(t_1)$, and the current velocity $y^v(t_1)$, and the previous input $u(t_1 - \Delta T)$, we calculate an approximate value of $p_j(t_1) = [\alpha_j(t_1), \beta_j(t_1)]^T$ where ΔT denotes the sampling interval. In other words, the inertial couplings at time t_1 among different joints are approximated with the input at time $t_1 - \Delta T$, and the nonlinear dependence of the manipulator dynamics on q, $\dot{q}$ is taken into consideration by using manipulator's actual behavior (i.e., its position and velocity). The latter implies that the approximation and update are made in *feedback form;* so is the resulting controller. Observe that $p_j(t_f) = [\alpha_j(t_f, \beta_j(t_f)]^T$ can be determined *a priori* by using the given final target position and velocity. Then, the arithmetic average[3] of these values at time t_1

$$\bar{p}_j(t_1) = \frac{p_j(t_1) + p_j(t_f)}{2} \quad (20)$$

is used to determine the optimal control input at time t_1, u_j^*, using the switching curves shown in Figs. 1–3.

The averaged dynamics method is illustrated in Fig. 4. At time $t_1 \in [0, t_f]$, the optimal control input should be determined based on the information of parameter function $p_j(t)$, for $t_1 \leq t \leq t_f$. However, the parameter values are known for both the current state, $p_j(t_1)$, and the final state, $p_j(t_f)$, but unknown for $t_1 < t < t_f$. Since the switching times in the optimal control are determined on the basis of the dynamic equations in $[t_1, t_f]$, it is necessary to find an approximate value of the parameter which represents the behavior of the manipulator during the entire remaining time period, i.e., $[t_1, t_f]$. In order to obtain such an approximate value of the parameter vector, $\bar{p}_j(t_1)$, we used the two known values of the parameter vector, i.e., $p_j(t_1)$, and $p_j(t_f)$, and made a zeroth order approximation of the parameter vector function, $p_j(t)$, for t_1

[3] Acutally, this does not have to be strictly an arithmetic average. A more general form would be $\bar{p}_j(t_1) = \eta p_j(t_1) + (1 - \eta)p_j(t_f)$ where $0 \leq \eta \leq 1$.

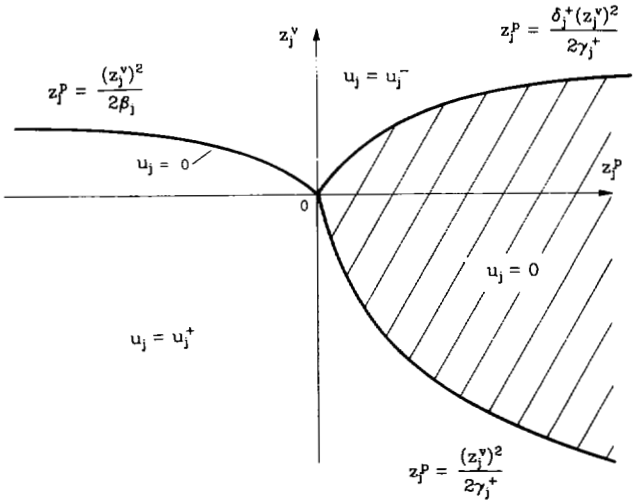


Fig. 3. Switching curves when $-\beta > \lambda$.

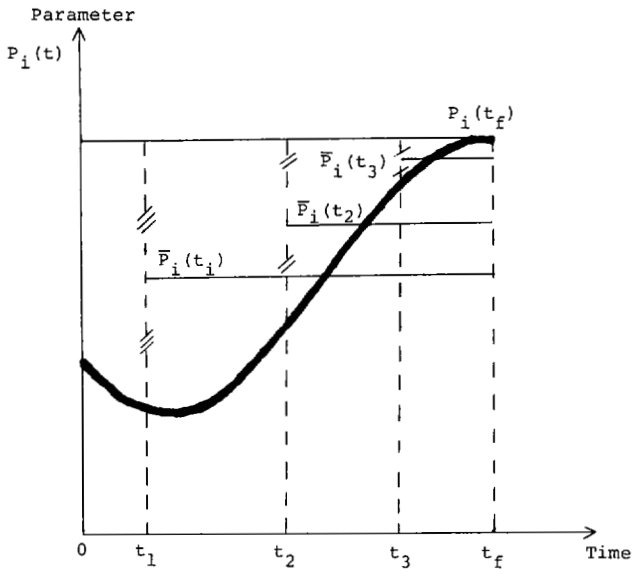


Fig. 4. The averaged dynamics method.

$\leq t \leq t_f$. The same procedure is repetitively applied for $t = t_2$, t_3, $\cdots$. It is apparent from Fig. 4 that $\bar{p}_j(t) \rightarrow p_j(t_f)$ as $t \rightarrow t_f$.

The averaged dynamics method takes advantage of the fact that the optimal control is dependent on the accumulation of nonlinear dynamics from the current state to the final state, and we know the coefficients of the manipulator dynamics for the current state and the final state. The modeling error in the averaged dynamics is also implicitly compensated at the time of the next update through position and velocity feedbacks which are used in the averaging process.

C. Synchronization of Final Time

If we use the same value for all λ_j's in local, joint NMTF controllers, the final times T_j, $j = 1, 2, \cdots, n$ may not be the same for all joints. Hence, the synchronization of all joints with the same final time T is achieved by adjusting λ_j for each joint j as follows.

First, using the switching curves in state space for each joint, we can obtain final time T_j for each joint j as a function of the initial position and velocity as well as the parameters for the jth joint of the manipulator. For example, if the initial state (x_{j0}^p, x_{j0}^v) is in the region $u_j = u_j^+$ in Fig. 1, then the control input sequence

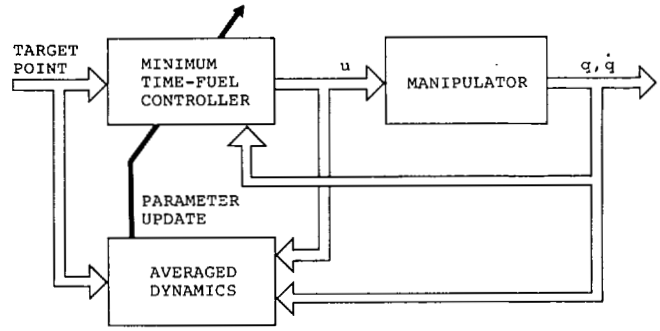


Fig. 5. Block diagram of the near-minimum time-fuel controller.

will be $u_j^+ \rightarrow 0 \rightarrow u_j^-$ and the final time T_j is computed to be (see the Appendix for derivation of this equation):

$$T_j = -\frac{x_{j0}^v}{\bar{\gamma}_j^+} + [\bar{\alpha}_j(\lambda_j - \bar{\beta}_j) + 2]\sqrt{\frac{(x_{j0}^v)^2/\bar{\gamma}_j^+ - 2x_{j0}^p}{\bar{\alpha}_j(\lambda_j - \bar{\beta}_j)^2 + (4 - 4\bar{\beta}_j/\bar{\gamma}_j^-)\lambda_j}} \tag{21a}$$

where

$$\bar{\alpha}_j = 1/\bar{\gamma}_j^+ - 1/\bar{\gamma}_j^-$$
$$\bar{\gamma}_j^+ = \bar{\alpha}_j u_j^+ + \bar{\beta}_j$$
$$\bar{\gamma}_j^- = \bar{\alpha}_j u_j^- + \bar{\beta}_j.$$

If the initial state is in the region $u_j = u_j^-$ in Fig. 1

$$T_j = -\frac{x_{j0}^v}{\bar{\gamma}_j^-} + [\bar{\alpha}_j(\lambda_j - \bar{\beta}_j) + 2]\sqrt{\frac{2x_{j0}^p - (x_{j0}^v)^2/\bar{\gamma}_j^-}{\bar{\alpha}_j(\lambda_j + \bar{\beta}_j)^2 + (4 - 4\bar{\beta}_j/\bar{\gamma}_j^+)\lambda_j}}. \tag{21b}$$

Similarly, we can compute T_j's for Figs. 2 and 3.

Then, choose which T_j is the longest, and let $T = \max\{T_j, j = 1, 2, \cdots, n\}$. This implies that all joints will be synchronized with the slowest. Therefore, for all but the slowest joint λ_j has to be so adjusted (reduced) that each axis may have the same final time T.

By arranging (21) with $T = T_j$, we get a second-order equation of λ_j for each case. Then, we can solve this second-order equation and validate its roots with (21) to obtain a unique λ_j for each joint.

Also note that ideally, λ_j's have to be recalculated whenever the switching curves are changed, requiring additional computation at each sampling interval. However, we can use their values at the first sampling interval for the entire control period as practical approximations for real-time implementation.[4]

D. Algorithm of the Near Minimum Time-Fuel Controller

Using 1) n local near minimum-time-fuel (NMTF) controllers for an n-jointed manipulator, 2) the synchronization method discussed above, and 3) the continual updating of the parameters with the average dynamics method, we can derive a global near minimum-time-fuel feedback controller as in Fig. 5, which is described below.

1) Given the initial and final states, compute the parameters $\bar{\alpha}_j(t_f)$, $\bar{\beta}_j(t_f)$, and compute λ_j, $j = 1, 2, \cdots, n$ in order to

[4] This may cause a slight asynchrony. But such an asynchrony is either negligible or can be avoided in practice by infrequent update (say, every five sampling periods) of λ_j's with a separate, parallel processor.

synchronize the final time for all joints by using (21). And, also set $k = 0$.

2) Calculate coefficients of the manipulator dynamics using the current state information.

3) For $j = 1, 2, \cdots, n$, perform Steps 4)–6).

4) Compute $\bar{p}_j(k) = [\bar{\alpha}_j(k), \bar{\beta}_j(k)]^T$, using (20).

5) Determine $u_j(k)$ using the switching curves described by (14)–(19).

6) If $|x_{jf} - x_j| < b_p$ and $|\dot{x}_{jf} - \dot{x}_j| < b_v$, then switch to a proportional + integral controller until the manipulator reaches the target point. Otherwise, $k = k + 1$, and go to step 2).

The NMTF controller is switched to a proportional + integral (PI) controller when the manipulator is within a prescribed range of the final target state

$$u_j(t) = \text{sat}\ (u_{\min j}, u_{\max j})[k_{pj}(x_{jf} - x_j) + \int_0^t \{k_{ij}^p(x_{if} - x_j) + k_{ij}^v(\dot{x}_{if} - \dot{x}_j)\}dt + k_{dj}(\dot{x}_{jf} - \dot{x}_j)]$$

where k_{pj}, k_{ij}^p, k_{ij}^v and k_{dj} [5] represent gains of the PI controller, and

$$\text{sat}\ (a_{\min}, a_{\max})a = \begin{cases} a_{\min} & \text{if } a \le a_{\min} \\ a & \text{if } a_{\min} < a < a_{\max} \\ a_{\max} & \text{if } a \ge a_{\max}. \end{cases}$$

The PI controller is employed because the bang-off-bang control is not desirable in the vicinity of the target state to prevent vibrations caused by frequent switchings of the control input, and therefore, the exact final state $(x_{jf}, \dot{x}_{jf})$ cannot be achieved. The bounds b_p and b_v should be selected by the designer to be large enough to prevent oscillatory behavior, but small enough to get a near-optimal solution.

The NMTF control proposed in this paper deviates from the exact optimal solution due to the following two reasons.

First, we used $\boldsymbol{u}(k-1)$ (i.e., $\boldsymbol{u}(t_1 - \Delta T)$ where t_1 belongs to the kth sampling interval) to 1) calculate $\beta_j(k)$ for $j = 1, 2, \cdots, n$, and 2) approximate the inertial coupling terms at time k. These parameter values are then used for determining optimal control at sampling time k. The use of old (by one sampling interval) control may introduce error in the calculation of the switching times. Let $\Delta \boldsymbol{u}(k) \equiv \boldsymbol{u}(k) - \boldsymbol{u}(k-1)$, $\Delta\alpha_j(k) \equiv \alpha_j(k) - \bar{\alpha}_j(k)$, and $\Delta\beta_j(k) \equiv \beta_j(k) - \bar{\beta}_j(k)$ for $j = 1, 2, \cdots, n$. Then, $\Delta \boldsymbol{u}(k)$ makes no contribution to $\Delta\alpha_j(k)$, since $\alpha_j(k)$ does not directly depend on $\boldsymbol{u}(k)$. This implies that no additional error due to the decoupling is introduced in the calculation of the parameter $\alpha_j(k)$ for all j. However, the contribution of $\Delta \boldsymbol{u}(k)$ to $\Delta\beta_j(k)$, denoted by $\Delta\beta_j^u(k)$, is defined as $\Sigma_{i=1, i\neq j}^n \boldsymbol{D}_{ji}^{-1}(\boldsymbol{q}(k))\, \Delta u_i(k)$. Except the case when switching in the control input $u_i(k)$ occurs, $\Delta u_i(k) = 0$, and therefore $\Delta\beta_j^u(k) = 0$. Therefore, error in the decoupling occurs only when switching in the control input takes place. Under normal operating conditions, there are usually two switchings in the manipulator control input between three different stages of control, namely, *acceleration, cruise,* and *deceleration.* Although it is difficult to analyze due to the impossibility of determining optimal switching times, the decoupling error should not be significant because of this small number of switchings.

Secondly, the NMTF control is based on the averaged dynamic equations, and thus may not be optimal for the overall nonlinear system. If we know the optimal solution $\boldsymbol{u}^*(t)$, $t_0 \le t \le t_f$, then we may be able to compare the performance indexes for both optimal and suboptimal solutions. However, we cannot analytically evaluate the suboptimality of the NMTF controller, since the exact optimal solution is impossible to determine. Note that the optimal switching curves, if they exist, have to be determined on the basis of the nonlinear dynamic behavior of the manipulator from the final state to intermediate state backward in time. Instead, we approximated the optimal switching curves for $t_{\text{current}} \le t \le t_f$ with the swtiching curves based on constant $\bar{p}(t)$ for $t_{\text{current}} \le t \le t_f$ which was determined by the average dynamics method. The value of $\bar{p}(t)$ is then updated—producing new switching curves—at the next iteration with the state feedback of the manipulator so that the error generated by the approximation at one iteration may be implicitly compensated at the subsequent iterations. This is a natural, linear time-varying approximation to the optimal nonlinear switching curves. Despite the impossibility of rigorous comparison between the optimal and the suboptimal solutions, this is a significant and novel departure from conventional manipulator controls in that, without sacrificing robot's capability and without requiring excessive computation, the NMTF can effectively handle the nonlinearity and joint couplings in the manipulator dynamics.

III. Computation Time Requirements

In this section we have explored the possibility of real-time implementation of the NMTF controller by analyzing its computation time requirements. The computation time requirements are one of the deciding factors for the sampling interval needed for real-time implementation. For the update of the controller parameters used for the switching curves, it is necessary to calculate the manipulator dynamic parameters $\boldsymbol{D}(\boldsymbol{q})$, and $\boldsymbol{f}(\boldsymbol{q}, \dot{\boldsymbol{q}})$, and the values of $\bar{\alpha}$, $\bar{\beta}$ where $\bar{\alpha} \equiv (\alpha_1, \cdots, \alpha_n)^T$ and $\bar{\beta} \equiv (\beta_1, \cdots, \beta_n)^T$. Using the Newton–Euler formulation of the manipulator dynamics with forward recursive equations, $\boldsymbol{D}(\boldsymbol{q})$ and $\boldsymbol{f}(\boldsymbol{q}, \dot{\boldsymbol{q}})$ can be computed with $O(n^2)$ multiplications and additions [16]. When the Jordan's method is employed, the inversion of $\boldsymbol{D}$ requires approximately $(n^3 - n^2)$ multiplications and $n(n-1)^2$ additions. The computation of $\bar{\alpha}$ and $\bar{\beta}$ also needs $O(n^2)$ computation, but with smaller coefficients. The switching curves require additional $O(n)$ computation. The total number of multiplications and additions required for the NMTF controller is summarized in Table I.

As can be seen from Table I, the major computational burden lies in the computation of the manipulator dynamic parameters, not in the computation of the control input. To investigate the feasibility of real-time implementation of the NMTF controller, we evaluated the required computation time for a specific example computer PDP-11/45, with a floating point processor FP 11-C. The floating point processor works in parallel with the central processor and can compute a floating point addition in 0.95 μs and a floating point multiplication in 2.52 μs both for 32 bit standard data. For the case of typical six degrees of freedom manipulators, it requires 1794 multiplications and 1806 additions, resulting in a total computation time of 6.2 ms.[6] This figure represents only a specific hardware configuration; if we use faster floating point processors and/or use integer arithmetic with suitable scaling, the computation time can be reduced further. Also, it is apparent that the required computation time is expected to decrease as the related computer technology (particularly VLSI technology) advances.

IV. Simulation Results

We have performed numerical simulation of the NMTF controller on DEC VAX11/780 with a dynamic model of the PUMA 600 series manipulator. The manipulator is manufactured

[5] Since the state vector consists of both position and velocity, the first and the last terms within [ ] of this equation can be regarded as proportional control. So, the entire control remains to be the PI control.

[6] This figure should not be compared to the computation times required for various computed torque methods, since the NMTF controller is totally different from them.

TABLE I
COMPUTATION TIME REQUIREMENTS

Variables	Multiplications	Additions
$\mathbf{D}, \mathbf{f}$	$21\,n^2 + 117\,n$	$28\,n^2 + 88\,n$
$\mathbf{D}^{-1}$	n^3-n^2	$n(n-1)^2$
$\bar{\alpha}, \bar{\beta}$	$n^2 + 3\,n$	$n^2 + 3\,n$
$\mathbf{u}$	$17\,n$	$11\,n$
Total	$n^3 + 21\,n^2 + 137\,n$	$n^3 + 27\,n^2 + 103\,n$

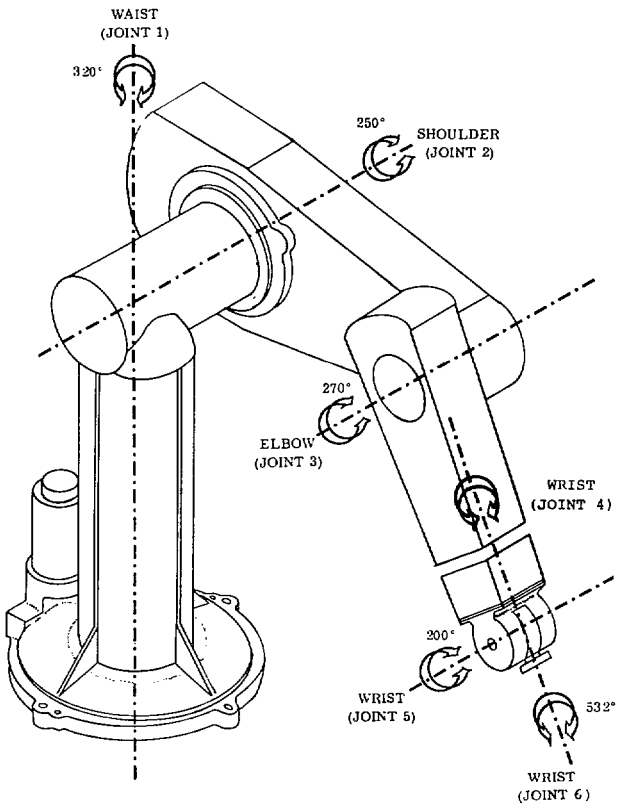


Fig. 6. Schematic diagram of the PUMA 600 manipulator.

TABLE II
LINK COORDINATE SYSTEM FOR A PUMA 600 ROBOT

Joint i	θ_i	d_i (mm)	a_i (mm)	α'_i (deg.)	Range (deg.)
1	θ_1	0	0	-90	-160 to 160
2	θ_2	149.5	432	0	-225 to 45
3	θ_3	0	0	90	-45 to 225

by Unimation, Inc., and consists of six rotational joints, each of which is driven by a dc servomotor. (See Fig. 6 for a schematic diagram of this manipulator.)

Using the Denavit and Hartenberg representation, the orthonormal link coordinate system for the PUMA 600 manipulator is given in Table II. We employed the Lagrangian formulation to derive the PUMA manipulator dynamics as in (22), which is then used to simulate the behavior of the first three joints of the manipulator.

$$\sum_{i=1}^{n} d_{ji}(q)\ddot{q}_i + \sum_{i=1}^{n}\sum_{k=1}^{n} h_{jik}(q)\dot{q}_i\dot{q}_k + g_j(q) = u_j, \qquad j = 1, 2, \cdots, n. \tag{22}$$

Typical terms have the following form:

$$\begin{aligned} d_{11}(q) &= J_{111} + J_{133} + J_{233} + 2J_{234}d_2 + J_{244}d_2^2 + J_{322} + J_{333} + J_{334}d_2^2 \\ &\quad + (J_{211} + 2J_{214}a_2 + J_{244}a_2^2 + J_{344}a_2^2)C_2^2 + J_{222}S_2^2 \\ &\quad + (J_{311} - J_{333})C_{23}^2 + 2J_{334}a_2C_2S_{23} \end{aligned}$$

$$\begin{aligned} h_{211} &= (J_{211} + J_{222} + 2J_{214}a_2 + J_{244}J_{244}S_2 + J_{344}a_2^2)S_2C_2 \\ &\quad + (J_{311} - J_{333})C_{23}S_{23} - J_{334}a_2(C_2C_{23} - S_2S_{23}) \end{aligned}$$

$$g_2 = -(a_2m_3 + a_2m_2 + m_2\bar{x}_2)C_2g - m_3\bar{z}_3S_{23}g$$

where J_{jik} is the (i, k)th element of the 4×4 inertia tensor, J_j, for the jth joint; d_2, a_2 are lengths related to the arm coordinate frame; $(\bar{x}_i, \bar{y}_i, \bar{z}_i)$ represents the center of mass for link i; and for i, $j = 1, 2, 3$, $C_j = \cos(q_j)$, $S_j = \sin(q_j)$, $C_{ji} = \cos(q_j + q_i)$, $S_{ji} = \sin(q_j + q_i)$.

The remaining three joints are used mainly for orienting the end-effector, but they are not considered here for simplicity. Note that this simulator for the forward dynamics of the manipulator is not part of the controller and will be replaced by the manipulator in case of acutal implementation.

The controller computes first the parameters associated with both the initial and the final states. At each sampling time, the controller computes the current values of the manipulator dynamic parameters, and then determines the parameters for the next iteration with the averaging between the current and final values. Control input is then determined using the switching curves as described in (14)–(19). The switching curves are constructed on the basis of the parameters updated.

Numerical values used in this simulation are as follows.

1) The mass, center of mass, and inertia of each joint of the PUMA manipulator are given in Table III.

2) The bounds on the control input torque are assumed to be

$$-300 \text{ nm} \le u_1 \le 300 \text{ nm}$$

$$-400 \text{ nm} \le u_2 \le 400 \text{ nm}$$

$$-200 \text{ nm} \le u_3 \le 200 \text{ nm}.$$

Observe that the choice of these parameters is made almost arbitrary for the sake of numerical demonstration and any of such choices does not change the basic performance of our manipulator control method.

In order to examine the performance of the NMTF controller for different ranges of motion, the first three joints of the manipulator are commanded to move from various initial points to final points. The corresponding simulation results are given in Table IV and show that the NMTF controller performs well for a wide range of motions. Note that a sampling period of 1 ms is used in all simulations except the ones with results in Table VI. Particularly, the results indicate the relative insensitivity to the range of motion and are therefore in sharp contrast with those reported in [1] which showed 20 percent or more overshoots for a small range of motion.

For the case of no load Fig. 7(a) shows the response of each joint of the manipulator to the NMTF controller command, and Fig. 7(b) the corresponding control input applied to each joint.

TABLE III
MASS, FIRST MOMENTS, AND INERTIAS OF THE FIRST THREE LINKS FOR THE PUMA 600 MANIPULATORS

Link	Mass	Center of Mass			Inertia		
	M (Kg)	$\bar{x}$ (m)	$\bar{y}$ (m)	$\bar{z}$ (m)	I_x ($Kg\,m^2$)	I_y ($Kg\,m^2$)	I_z ($Kg\,m^2$)
1	2.27	0.0	0.0	0.075	0.00376	0.00376	0.0169
2	15.91	-0.216	0.0	0.0	0.9897	0.1237	0.1237
3	11.36	0.0	0.0	0.216	0.0074	0.0074	0.7067

TABLE IV
SIMULATION RESULTS WITH VARIOUS INITIAL AND FINAL CONDITIONS

Case		I	II	III
Initial condition (deg)	$q_1(0)$	60.0	45.0	-45.0
	$q_2(0)$	-80.0	-30.0	45.0
	$q_3(0)$	80.0	150.0	60.0
Final condition (deg)	$q_1(t_f)$	40.0	0.0	0.0
	$q_2(t_f)$	-90.0	-60.0	-45.0
	$q_3(t_f)$	90.0	210.0	150.0
Settling time (sec)	T	0.11	0.19	0.31
Overshoot (%)	Joint 1	0.7	5.0	5.2
	Joint 2	6.5	0.0	5.9
	Joint 3	3.7	1.3	6.1

The first three joints of the manipulator are now ordered to move from initial joint angles (45°, −30°, 150°) to final joint angles (0°, −60°, 210°). Fig. 8 presents the case when the manipulator picks up a maximum load (i.e., 5 lb ≈ 2.25 kg), demonstrating the ability that, even with a maximum payload, the controller can drive the manipulator with high performance. This result is not unexpected since the precise parameters and the load characteristics for the manipulator are assumed to be known to the NMTF controller. If these are unknown, one has to appeal to an adaptive control method similar to those in Dubowsky and DesForges [13], Koivo and Guo [14], and Kim and Shin [15]. Table V summarizes the simulated effects on overshoot in each joint when the weighting factor λ is varied. It indicates that as the weighting factor increases, the NMTF controller approaches a near time-optimal controller, and tends to have a slight increase in overshoot.

The effects of the variation of the sampling interval on overshoot are simulated and presented in Table VI. Since the dynamic model used for the NMTF controller is updated at every sampling interval, these effects can be employed to measure implicitly the error induced by both the averaging process and the discrete implementation on a digital computer. The sampling interval is varied from 0.001 s to 0.02 s and the result shows that as the sampling interval increases, the overshoot increases accordingly. This result is expected; the NMTF controller is derived for continuous-time domain with the averaged dynamic equations, and, therefore, it would become more accurate when the sampling interval gets smaller. In general, it is desirable to select the sampling interval which 1) provides acceptable/required accuracy, and 2) does not require excessive computation.

Particularly, when the sampling interval is varied, the NMTF

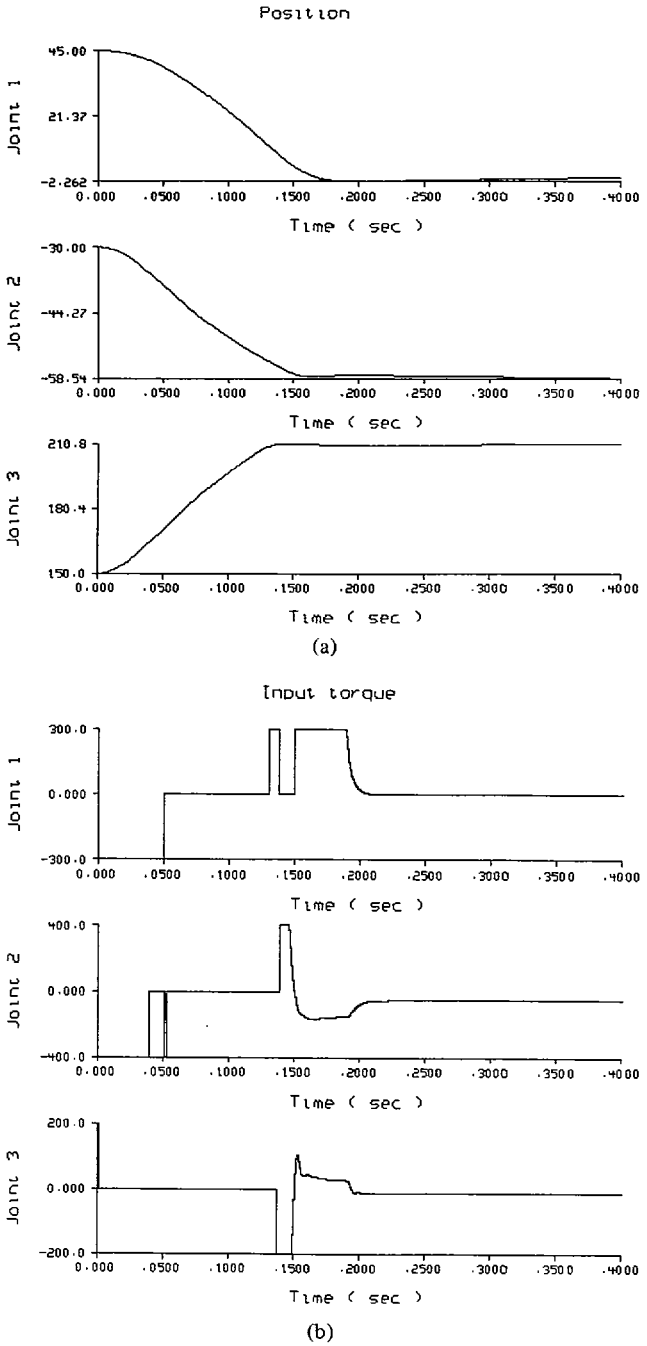


Fig. 7. (a) Manipulator responses with no load. (b) Control input with no load.

control offers reasonably favorable performance as long as the sampling interval is kept close to or below 5 ms. This result indicates that the NMTF controller can be implemented in real-time with a PDP-11/45 or similar computers as evidenced in the previous computation time analysis.

Our simulation results have indicated that the manipulator can be driven at high speed with reasonable accuracy regardless of the load it is carrying (see Table IV and Fig. 8). One can observe that since our NMTF controller is developed on the basis of the averaged dynamics method, it is not restricted to any range of motion (unlike the one in [1]). In addition to high operation speed, this controller has a possibility of real-time implementation with a suitable choice of sampling and update intervals. As a whole, this simulation exhibits that the proposed control method has great

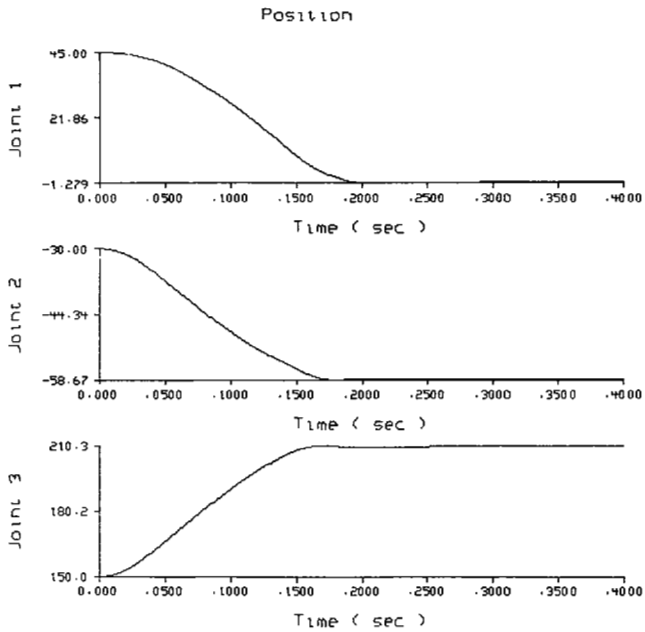


Fig. 8. Manipulator responses with full load.

TABLE V
EFFECTS OF THE WEIGHTING FACTOR λ ON OVERSHOOT (%)

λ	Joint 1	Joint 2	Joint 3
10	2 2	0.0	0.8
100	5.0	0.0	1.3
1000	6.9	3.3	2.0
100,000	12 2	3.3	16.0

TABLE VI
EFFECTS OF SAMPLING INTERVAL (ΔT) ON OVERSHOOT (%)

ΔT	Joint 1	Joint 2	Joint 3
0.001	5.0	0.0	1.3
0.002	5.0	0.0	3.9
0.005	12.6	0.0	2.2
0.01	22.1	12.0	27.7
0.02	25.2	22.0	29.7

potential for high performance with the capability of coping with the nonlinear, coupled dynamics of the manipulator.

V. CONCLUSION

We have presented the design of a manipulator feedback control system on the basis of the minimum time-fuel criterion, which has a direct link to the goal of improving productivity and saving energy. The design has focused on the following important features. Unlike most other existing methods, this control method is not based on a conventional, separate path planning algorithm which does not take the manipulator dynamics into account and therefore results in the possible underutilization of manipulator's capability. The design of the NMTF controller has placed emphasis, particularly on the improvement of the manipulator performance by nearly fully utilizing the manipulator's capability. This was made possible with a judicious approximation with the averaging of the manipulator dynamics. This controller provides a feedback control algorithm with a simple updating structure so that it can be implemented in real-time on mini- or microcomputers. Primarily, this controller is intended to provide fast operation speed with less energy as well as reasonable settling accuracy.

The averaged dynamics method is a new solution to one of the most difficult problems in robot control, namely the nonlinearity and joint couplings in the manipulator dynamics. This is a simple yet novel departure from conventional robot control techniques toward the goal of efficient control of the manipulator.

It should be noted that the NMTF controller derived in this paper is concerned only with free-space motion (i.e., noncompliant motion). For the compliant case, the dynamic equation becomes far more complex; for example, the static friction terms cannot be ignored, but cannot be modeled accurately, either. With an accurate dynamic model of the manipulator for the compliant case, and with a suitable performance criterion[7] an optimal controller could be derived. As of this writing, this is an unsolved, challenging problem in robot control.

In short, the manipulator control method proposed here has great potential use—due to its high performance capability and simplicity in structure—in the design of intelligent, noncompliant controllers for the growing number of sophisticated industrial manipulators.

APPENDIX

DERIVATION OF (21a)

T_j is the sum of three components: 1) t_{aj}, the traveling time from initial state $x_0 = (x_{j0}^p, x_{j0}^v)^T$ to a point a on the switching curve $z_j^p = \bar{\delta}_j^- (z_j^v)^2/2\bar{\gamma}_j^-$ with control input $u_j = u_j^+$, 2) t_{bj}, the traveling time from the point A to a point B on the switching curve $z_j^p = (z_j^v)^2/2\bar{\gamma}_j^-$ with control input $u_j = 0$, and 3) t_{cj}, the traveling time from the point B to the origin in the state space with $u_j = u_j^-$.

Then, one can get

$$t_{aj} = \frac{x_j^v(A) - x_{j0}^v}{\bar{\gamma}_j^+} \tag{A.1}$$

$$t_{bj} = \frac{x_j^v(B) - x_j^v(A)}{\bar{\beta}_j} \tag{A.2}$$

$$t_{cj} = \frac{x_j^v(A)}{\bar{\gamma}_j^-} . \tag{A.3}$$

We obtain $x_j^v(B)$ as a function of $x_j^v(A)$,

$$x_j^v(B) = \frac{\lambda_j + \bar{\beta}_j}{\lambda_j - \bar{\beta}_j} x_j^v(A) \tag{A.4}$$

and then $x_j^v(A)$ as a function of the initial condition x_{j0}^v, x_{j0}^p

$$x_j^v(A) = \sqrt{\frac{(x_{j0}^v)^2/\bar{\gamma}_j^+ - 2x_{j0}^p}{\bar{\alpha}_j + (4 - 4\bar{\beta}_j/\bar{\gamma}_j)\lambda_j/(\lambda_j - \bar{\beta}_j)^2}} . \tag{A.5}$$

Combining these equations, we can finally obtain (21a).

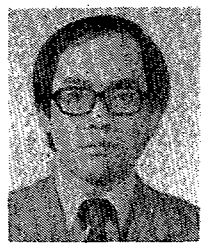
[7] Note that the minimum time fuel criterion may not have much practical significance in the compliant case.

ACKNOWLEDGMENT

The authors wish to thank N. D. McKay and anonymous reviewers for constructive comments on the material presented in this paper.

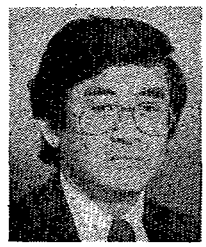
REFERENCES

[1] M. E. Kahn and B. E. Roth, "The near-minimum-time control of open-loop articulated kinematic chains," *ASME J. DSMC,* vol. 93, no. 3, pp. 164–172, Sept. 1971.
[2] C. S. Lin, P. R. Chang, and J. Y. S. Luh, "Formulation and optimization of cubic polynomial joint trajectories for mechanical manipulators," in *Proc. IEEE Conf. Decision Contr.,* Orlando, FL, Dec. 1982.
[3] J. Y. S. Luh and C. S. Lin, "Optimum path planning for mechanical manipulators," *ASME J. DSMC,* vol. 102, pp. 142–151, June 1981.
[4] J. Y. S. Luh, M. W. Walker, and R. P. C. Paul, "Resolved-acceleration control of mechanical manipulators," *IEEE Trans. Automat. Contr.,* vol. AC-25, pp. 468–474, June 1980.
[5] J. Y. S. Luh, M. W. Walker, and R. P. C. Paul, "On-line computational scheme for mechanical manipulators," *ASME J. DSMC,* vol. 102, pp. 69–76, June 1980.
[6] P. M. Lynch, "Minimum-time, sequential axis operation of a cylindrical, two-axis manipulator," in *Proc. Joint Automat. Contr. Conf.,* 1981, vol. 1, paper WP-2A.
[7] W. E. Snyder and W. A. Gruver, "Microprocessor implementation of optimal control for a robotic manipulator system," in *Proc. IEEE Conf. Decision Contr.,* 1979, pp. 839–841.
[8] T. L. Turner and W. A. Gruver, "A viable suboptimal controller for robotic manipulators," in *Proc. IEEE Conf. Decision Contr.,* Dec. 1980, pp. 83–87.
[9] D. E. Whitney, "Resolved motion rate control of manipulators and human prosthesis," *IEEE Trans. Man-Machine Syst.,* vol. MMS-10, pp. 47–53, June 1969.
[10] J. Y. S. Luh, "An anatomy of industrial robots and their controls," *IEEE Trans. Automat. Contr.,* vol. AC-28, pp. 133–153, Feb. 1983.
[11] K. G. Shin and S. M. Malin, "Dynamic adaptation of robot kinematic control to its actual behavior," in *Proc. 1981 IEEE Int. Conf. Cybern. Soc.,* Atlanta, GA, Oct. 1981.
[12] A. P. Sage, *Optimum Systems Control.* Englewood Cliffs, NJ: Prentice-Hall, 1968.
[13] S. Dubowsky and D. T. DesForges, "The application of model-referenced adaptive control to robot manipulators," *ASME J. DSMC,* vol. 101, pp. 193–200, Sept. 1979.
[14] A. J. Koivo and T. H. Guo, "Adaptive linear controller for robotic manipulators," *IEEE Trans. Automat. Contr.,* vol. AC-28, pp. 162–171, Feb. 1983.
[15] B. K. Kim and K. G. Shin, "Adaptive model following control of robotic manipulators," *IEEE Trans. Aerosp. Electron. Syst.,* vol. AES-19, pp. 805–814, Nov. 1983.
[16] V. Cvetkovic and M. Vukobratovic, "Computer oriented algorithm for modeling active spatial mechanism for robotics applications," *IEEE Trans. Syst. Man, Cybern.,* pp. 838–847, Nov. 1982.
[17] E. C. Beattie *et al.,* "Sensor failure detection system," Tech. Rep. NASA CR-165515, Aug. 1981.
[18] R. L. DeHoff and S. M. Lock, "Development of simplified nonlinear models from multiple linearizations," in *Proc. IEEE Conf. Decision Contr.,* 1978.



Byung Kook Kim (S'75–M'81) was born in Choongju, Korea, on October 5, 1952. He received the B.S. degree in electronics engineering from Seoul National University, Seoul, Korea, in 1975 and the M.S. and the Ph.D. degrees in electrical science from the Korea Advanced Institute of Science and Technology (KAIST), Seoul, Korea, in 1977 and 1981, respectively.

From 1981 to 1982, he was a Research Engineer at Woojin Instrument Corporation, Seoul, Korea. From 1982 to 1983 he was a Visiting Scholar at Rensselaer Polytechnic Institute, Troy, NY, and then at the University of Michigan, Ann Arbor, conducting research in robotics. He is currently back at Woojn Instrument Corporation, where his fields of current interest include computer control systems, time-delay systems, and robotics.



Kang G. Shin (S'75–M'78–SM'83) was born in the province of Choongbuk, Korea, on October 20, 1946. He received the B.S. degree in electronics engineering from Seoul National University, Seoul, Korea, in 1970, and both the M.S. and Ph.D. degrees in electrical engineering from Cornell University, Ithaca, NY, in 1976 and 1978, respectively.

From 1978 to 1982 he was an Assistant Professor at Rensselaer Polytechnic Institute, Troy, NY. He was also a Visiting Scientist at the U.S. Air Force Flight Dynamics Laboratory in the summer of 1979 and at Bell Laboratories, Holmdel, NJ, in the summer of 1980 where his work was concerned with distributed airborne computing and cache memory architecture, respectively. He taught short cources for the IBM Computer Science Series in the area of computer architecture. Since September 1982, he has been with the Department of Electrical Engineering and Computer Science, University of Michigan, Ann Arbor, where he is currently an Associate Professor. His current teaching and research interests are in the areas of robotics and automation, distributed real-time computing, and computer architecture.

Dr. Shin is a member of the Association for Computing Machinery, Sigma Xi, and Phi Kappa Phi.