[18] ——, "Disaggregate clustering using the concept of mutual nearest neighborhood," *IEEE Trans. Syst., Man, Cybern.*, vol. SMC-8, pp. 888–895, Dec. 1978.
[19] ——, "The condensed nearest neighbour rule using the concept of mutual nearest neighborhood," *IEEE Trans. Inform. Theory*, vol. IT-25, pp. 488–490, July 1979.
[20] ——, "Learning with a mutualistic teacher," *Pattern Recog.*, vol. 11, pp. 383–390, 1979.
[21] Y. Kodratoff, *Introduction to Machine Learning*. London: Pitman, 1988.
[22] M. Ichino, "Pattern classification based on the Cartesian join system: A general tool for feature selection," *Proc. IEEE Int. Conf. Syst., Man, Cybern.*, Atlanta, GA, Oct. 14–17, 1986.
[23] R. Hamming and E. Gilbert, "Probability of ocurrence of a constant number of isolated pairs in a random population," *Univ. Wisconsin Computing News*, no. 32, 1954.
[24] R. O. Duda and P. E. Hart, *Pattern Classification and Scene Analysis*. New York: Wiley Interscience, 1973.
[25] P. M. Forget, J. Lebb, H. Puig, R. Vignes, and M. Hideux, "Microcomputer-aided identification: An application to trees from French Guiana," *Botanical J. Linnean Soc.*, vol. 93, pp. 01–018, 1986.

An Approach to Robot Motion Analysis and Planning for Conveyor Tracking

T. H. Park and B. H. Lee

Abstract— **The robot motion is analyzed and planned for conveyor tracking considering the speed of the conveyor belt and the locations of the part and the robot. The joint torque limit, together with the joint velocity, acceleration, and jerk limits of the robot, is also taken into account in the motion analysis and planning. To include the robot arm dynamics, the problem is formulated as the second-order state equations using a parametric function. The conveyor tracking problem is then converted to an optimal tracking problem. The solution that minimizes the specified performance index is obtained using the dynamic programming approach. Numerical examples are finally presented to demonstrate the significance of the proposed method for conveyor tracking of the robot in a workcell.**

I. INTRODUCTION

The robotic workcell with robot manipulators and conveyor belt systems is desirable in a broad class of industrial applications. In most of the factory automation system, the part is transferred by the conveyor belt system. The primary advantage of conveyor-tracking applications is that continuous operations may maximize robot utilization and minimize production cycle time, contributing to economic performance and system productivity [1]. Efficient control of a robot manipulator for conveyor tracking, however, has several problems as follows.

1) The steady-state tracking error should be eliminated since it is closely related to the task accuracy. Manipulation tasks are normally carried out in the steady state where the relative position vector from the robot hand to the part becomes the zero vector.
2) Since the position of the robot hand is different from that of the part in the initial state, there should be the transient state until the motion of the robot hand becomes equal to that of the part. The velocity of the conveyor belt system, together with the initial positions of the robot hand and the part, has important effects on the motion in the transient state.
3) The physical constraints of the robot manipulator must be taken into account in robot motion planning. The tracking trajectories should be generated subject to the torque constraints of the manipulator and the smoothness constraints of the trajectory [11]. In order to maximize the task productivity, the settling time of the tracking trajectory must be minimized subject to the constraints.

The purpose of this paper is to analyze and plan the motion of the robot manipulator for tracking the conveyor belt system subject to the various requirements mentioned previously. We first consider the robot control structure. The control structure of the robot is generally divided into two hierarchical levels. The lower level includes path tracking control, and the upper level includes motion planning of the manipulator. Due to the division of the control structure, the highly coupled nonlinear dynamics can be easily included in the upper level. [9] Since the nonlinearity of the manipulator dynamics is taken into account at this level, the tracking controller can generally keep the manipulator fairly close to the desired trajectory.

We next consider the conveyor belt system coupled with a robot manipulator. Several robot-conveyor belt systems have been announced so far. In the system developed by Holland *et al.* [2] and Mo *et al.* [3], the robot tracks the moving part on the conveyor belt system and the tracking motion is planned using the method of the straight line trajectory planning, presented by Paul [6] and Taylor [7]. However the robot motion was planned without considering the arm dynamics, while the conveyor belt speed and the initial position of the robot were considered in tracking motion. Also, there are other control methods taking the arm dynamics into account in the motion planning stage such as the minimum time control problem [8]–[11].

Here, we want to incorporate the robot arm dynamics together with the various requirements in the error-free conveyor tracking problem. We formulate the problem as an optimal tracking problem in the optimal control theory. The robot arm dynamics will be used in obtaining the second-order state equations. The limit of the joint torque will be considered as the bound on the control variables. Also the limit of the joint velocity, acceleration, and jerk will be accommodated as the constraints on the state variables. An optimal solution, which minimizes the performance index subject to the various constraints, is then obtained through dynamic programming. This paper is organized as follows. In Section II, the conveyor tracking problem is stated with some assumptions, and the problem formulation is presented in Section III. In Section IV, a dynamic programming algorithm is presented to analyze and plan the robot arm motion. Finally, we discuss simple examples to demonstrate the utility of the proposed motion planning algorithm with some simulation results.

II. A CONVEYOR TRACKING PROBLEM

We consider the system model consisting of a conveyor belt system and a nonredundant robot manipulator that has n degrees of freedom. The system model is assumed to have the following characteristics:

Manuscript received October 6, 1990; revised August 13, 1991. A portion of this paper was presented at the 1991 IEEE International Conference on Robotics and Automation, Sacramento, CA, April 9–11, 1991.

The authors are with the Department of Control and Instrumentation Engineering, Automation and Systems Research Institute, and Engineering Research Center for Advanced Control and Instrumentation, Seoul National University, Seoul 151–742, Korea.

IEEE Log Number 9104561.

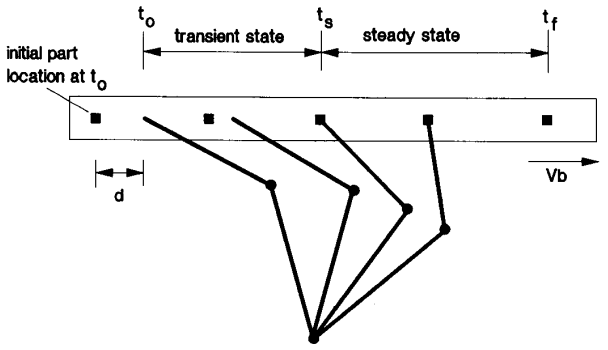


Fig. 1. Problem formulation for conveyor tracking.

1) The conveyor belt system runs at the constant velocity, and the part is stationary with respect to the conveyor belt system.
2) The orientation of the robot hand is aligned with the part initially and kept fixed during the robot tracks the part.
3) The robot tracks the part along a straight line path over the conveyor belt system.

The problem formulation for conveyor tracking is illustrated in Fig. 1. Since the conveyor belt system is assumed to run in a constant velocity, the position and velocity of the part on the conveyor belt system is described as

$$\boldsymbol{x_p}(t) = \boldsymbol{\nu_b} t - \boldsymbol{\nu_b} t_o + \boldsymbol{x_p}(t_o), \qquad t \geq t_o \tag{1}$$

$$\dot{\boldsymbol{x}}\boldsymbol{p}(t) = \boldsymbol{\nu_b}, \qquad t \geq t_o \tag{2}$$

where $\boldsymbol{\nu_b}$ denotes the velocity vector of the conveyor belt, and $\boldsymbol{x_p}(t)$ denotes the position vector of the part with respect to a reference frame at time t. On the other hand, at the initial time, the robot hand is at a standstill on the forecasting path of the part, that is

$$\boldsymbol{x_h}(t_o) = \boldsymbol{x_p}(t_o + \boldsymbol{d}) \tag{3}$$

$$\dot{\boldsymbol{x}}_{\boldsymbol{h}}(t_o) = \boldsymbol{0} \tag{4}$$

where $\boldsymbol{x_h}(t)$ denotes the position vector of the robot hand with respect to a reference frame at time t, and $\boldsymbol{d}$ denotes the deviation vector between the initial positions of the part and the robot hand.

The robot is required to match the position, velocity and acceleration of its hand with those of the part, and the overall motion is divided into two states: the transient state ($t_o \leq t \leq t_s$) in which the robot moves to reduce the error between the two trajectories, and the steady state ($t \geq t_s$) in which the error maintains to be below a threshold and some desired operations can be achieved with the part. The reaching time to the steady state is called as the settling time (t_s). It is notable that the settling time is not a given value but a value to be determined.

To maximize the usable performance of the manipulator on conveyor tracking, the robot motion to be planned has some requirements as follows.

1) *Steady state error:* The position and velocity error in the steady state may lead to lower accuracy of the task. Thus the trajectory for conveyor tracking must have the steady state error as small as possible.
2) *Torque and smoothness constraints:* The trajectory must be generated such that all the joint torques remain within their bounds for all time. The torque constraints are given as

$$u_{i,\min}(t) \leq u_i(t) \leq u_{i,\max}(t), \qquad i = 1, \cdots, n \tag{5}$$

where $u_i(t)$ is the ith element of the joint torque vector $\boldsymbol{u}(t) \in R^n$. Also the smoothness constraints are given as

$$V_{i,\min} \leq \dot{q}_i(t) \leq V_{i,\max}, \qquad i = 1, \cdots n \tag{6}$$

$$A_{i,\min} \leq \ddot{q}_i(t) \leq A_{i,\max}, \qquad i = 1, \cdots n \tag{7}$$

$$j_{i,\min} \leq q_i^{(3)}(t) \leq J_{i,\max}, \qquad i = 1, \cdots n \tag{8}$$

where $q_i(t)$is the ith element of the joint value vector $\boldsymbol{q}(t) \in R^n$, and $q_i^{(3)}(t)$ denotes the third derivative of $q_i(t)$.

3) *Settling time:* The necessary time for the task is reduced as the robot reaches to the steady state quickly. The settling time thus should be minimized considering the torque and smoothness constraints of the robot, the speed of the conveyor belt, and the initial positions of the robot and the part.

III. PROBLEM FORMULATION

A. Equations of Motion of a Robot Manipulator

To solve the conveyor tracking problem defined in the previous section, the complete dynamic model of a robot manipulator must be considered. In general, the dynamical behavior of an n joint manipulator can be described by the Lagrange–Euler equations of motion [17] such that

$$\boldsymbol{u}(t) = \boldsymbol{M}(\boldsymbol{q}(t))\ddot{\boldsymbol{q}}(t) + \boldsymbol{h}(\boldsymbol{q}(t), \dot{\boldsymbol{q}}(t)) \tag{9}$$

where $\boldsymbol{M}(\boldsymbol{q}(t)) : R^n \rightarrow R^{n \times n}$is the inertial matrix, and $\boldsymbol{h}(\boldsymbol{q}(t), \dot{\boldsymbol{q}}(t)) : R^n \times R^n \rightarrow R^n$ is the vector of gravitational, Coriolis and centrifugal forces. The state space representation of (9) requires $2n$-dimensional space. Since the robot hand moves along a straight line path between the $\boldsymbol{x_h}(t_o)$ and $\boldsymbol{x_h}(t_f)$, the position of the robot hand can be represented in the parameter form such that

$$\boldsymbol{x_h}(t) = \boldsymbol{x_h}(t_o) + r(t)(\boldsymbol{x_h}(t_f) - \boldsymbol{x_h}(t_o)) \tag{10}$$

where the $r(t) \in [0, 1]$ is a scalar function. In order to relate this parametric function to the joint values, the kinematical model of a robot manipulator should be taken into account. For the class of nonredundant manipulators it is possible to obtain the inverse kinematic solution. Since the orientation of the robot hand is assumed to be kept fixed during the robot tracks the part, the inverse kinematic relations can be written as

$$\boldsymbol{q} = \boldsymbol{f}(r(t)) \tag{11}$$

$$\dot{\boldsymbol{q}}(t) = (d\boldsymbol{f}(r(t))/dr(t))\dot{r}(t) \tag{12}$$

$$\ddot{\boldsymbol{q}}(t) = (d\boldsymbol{f}(r(t))/dr(t))\ddot{r}(t) + (d^2\boldsymbol{f}(r(t))/dr(t)^2)\dot{r}(t)^2 \tag{13}$$

where $\boldsymbol{f}(\cdot) : R \rightarrow R^n$ is a function that depends on the configuration of the manipulator. Substituting (11)–(13) into (9) yields

$$\begin{aligned}\boldsymbol{u}(t) = {} & \boldsymbol{M}(r(t))(d\boldsymbol{f}(r(t))/dr(t))\ddot{r}(t) \\ & + \boldsymbol{M}(r(t))\left(d^2\boldsymbol{f}(r(t))/dr(t)^2\right)\dot{r}(t)^2 \\ & + \boldsymbol{h}(r(t), \dot{r}(t))\end{aligned} \tag{14}$$

and a single differential equation can be obtained by multiplying the n equations in (14) by $[\boldsymbol{M}(r(t))d\boldsymbol{f}(r(t))/dr(t)]^T$, that is

$$\begin{aligned}&[\boldsymbol{M}(r(t))\ d\boldsymbol{f}(r(t))/dr(t)]^T\boldsymbol{u}(t) = \\ &\quad [\boldsymbol{M}(r(t))\ d\boldsymbol{f}(r(t))/dr(t)]^T[\boldsymbol{M}(r(t))\ d\boldsymbol{f}(r(t))/dr(t)]\ddot{r}(t) \\ &\quad + [\boldsymbol{M}(r(t))\ d\boldsymbol{f}(r(t))/dr(t)]^T\left[\boldsymbol{M}(r(t))\ d^2\boldsymbol{f}(r(t))/dr(t)^2\right]\dot{r}(t)^2 \\ &\quad + [\boldsymbol{M}(r(t))\ d\boldsymbol{f}(r(t))/dr(t)]^T\boldsymbol{h}(r(t), \dot{r}(t)).\end{aligned} \tag{15}$$

Here, we introduce the state $\boldsymbol{x}(t)(= [x_1(t)\ x_2(t)]^T)$, where $x_1(t) \equiv r(t)$ and $x_2(t) \equiv \dot{r}(t)$. Since the $x_1(t)$ and $x_2(t)$ determine

the motion of the robot hand, the state $\boldsymbol{x}(t)$ is named the *hand state*. With the hand state, the differential equations in (15) become the second-order state equations such that

$$\dot{x}_1(t) = x_2(t) \tag{16a}$$

$$\dot{x}_2(t) = \frac{b(x_1(t))x_2(t)^2 + c(x_1(t), x_2(t))}{a(x_1(t))} + \sum_{i=1}^{n} d_i(x_1(t))u_i(t) \tag{16b}$$

where

$$a(x_1(t)) = [\boldsymbol{M}(x_1(t))\ d\boldsymbol{f}(x_1(t))/dx_1(t)]^T \cdot [\boldsymbol{M}(x_1(t))\ d\boldsymbol{f}(x_1(t))/dx_1(t)]$$

$$b(x_1(t)) = -[\boldsymbol{M}(x_1(t))\ d\boldsymbol{f}(x_1(t))/dx_1(t)]^T \cdot [\boldsymbol{M}(x_1(t))\ d^2\boldsymbol{f}(x_1(t))/dx_1(t)^2]$$

$$c(x_1(t), x_2(t)) = -[\boldsymbol{M}(x_1(t))\ d\boldsymbol{f}(x_1(t))/dx_1(t)]^T \boldsymbol{h}(x_1(t), x_2(t))$$

and

$$[d_1(x_1(t)), \cdots, d_n(x_1(t))] = \frac{[\boldsymbol{M}(x_1(t))\ d\boldsymbol{f}(x_1(t))/dx_1(t)]^T}{[\boldsymbol{M}(x_1(t))\ d\boldsymbol{f}(x_1(t))/dx_1(t)]^T [\boldsymbol{M}(x_1(t))\ d\boldsymbol{f}(x_1(t))/dx_1(t)]}.$$

The joint torques $u_i(t)(i = 1, \cdots, n)$ are used as the controls of the state equations. We now have the state space representation of the robot dynamic model, which associates the joint torques with the hand state.

B. An Optimal Tracking Problem

Finding an optimal control that let the state of the dynamic system track the desired trajectory, is under the category of an optimal tracking problem. Since the state of the dynamic equations in (16a) and (16b) specifies the motion of the robot hand, and since the trajectory of the part is the desired trajectory of the robot hand, the conveyor tracking problem can be formulated as an optimal tracking problem.

The trajectory of the part must also be represented by the parametric function such as that of the robot hand. Equation (10) means that the scalar function $r(t)$ can specify any position vector located on the straight line path between the $\boldsymbol{x}_h(t_o)$ and $\boldsymbol{x}_h(t_f)$. Since the part moves on this path, the position vector of the part in (1) can be described by the parametric function. Let us denote the $\widetilde{x_1}(t)$ and $\widetilde{x_2}(t)$ as the scalar functions that specify the position and velocity of the part, respectively. Define the notation $[\,]i$ as the ith element of a vector, then we have the *part state* $\tilde{\boldsymbol{x}}(t)$ $(= [\widetilde{x_1}(t)\widetilde{x_2}(t)]^T)$ as

$$\widetilde{x_1}(t) \equiv \frac{[\boldsymbol{\nu}_b t - \boldsymbol{\nu}_b t_o + \boldsymbol{x}_p(t_o) - \boldsymbol{x}_h(t_o)]_i}{[\boldsymbol{x}_h(t_f) - \boldsymbol{x}_h(t_o)]_i}, \qquad t \geq t_o \tag{17}$$

$$\widetilde{x_2}(t) \equiv \frac{[\boldsymbol{\nu}_b]_i}{[\boldsymbol{x}_h(t_f) - \boldsymbol{x}_h(t_o)]_i}, \qquad t \geq t_o \tag{18}$$

for some i such that $[\boldsymbol{x}_h(t_f) - \boldsymbol{x}_h(t_o)]_i \neq 0$. From the viewpoint of optimal tracking problem, the part state is the desired value of the hand state of the robot dynamic equations in (16a) and (16b). To bring the hand state closer to the part state, the performance index to be minimized is adopted as

$$\boldsymbol{J} = \int_{t_o}^{t_f} (x_1(t) - \widetilde{x_1}(t))^2 + (x_2(t) - \widetilde{x_2}(t))^2 dt. \tag{19}$$

The performance index becomes smaller as the difference between the hand state and the part state decreases in the time interval $[t_o, t_f]$, that is, the nearest approach of the robot hand to the part minimizes the specified performance index. However, if controls of the dynamic system are not bounded, minimizing (19) may result in the controls with impulses. The optimal controls that minimize the performance index, should be generated within the torque constraints in (5). Also, the smoothness constraints in (6)–(8) put limits on the hand state, which will be stated in Section IV. The boundary conditions of the hand state are given as

$$x_1(t_o) = 0, \qquad x_1(t_f) = \widetilde{x_1}(t_f) \tag{20}$$

$$x_2(t_o) = 0, \qquad x_2(t_f) = \widetilde{x_2}(t_f) \tag{21}$$

where $\widetilde{x_1}(t_f)$ and $\widetilde{x_2}(t_f)$ are determined by (17) and (18), respectively. We now have the mathematical conveyor tracking problem as follows. Given the velocity of the conveyor belt, and the initial positions of the part and the hand, find the hand state such that the performance index in (19) is minimized subject to the torque constraints in (5), the smoothness constraints in (6)–(8) and the boundary conditions in (20) and (21).

IV. SOLUTION TO THE PROBLEM

To solve the formulated conveyor tracking problem, we first adopt a proper method that can deal with the problem. Pontryagin's minimum principle is usually employed to obtain an optimal tracking solution as the closed form. Unfortunately, it is quite a difficult to apply the minimum principle to our problem, because of the nonlinearity of the dynamic equations and various constraints imposed on the states and controls. On the other hand, algorithmic approaches are able to overcome such difficulties and these can be implemented through off-line motion planning. Shin *et al.* [9], Bobrow *et al.* [10] and Geering *et al.* [8] proposed specific methods to plan the minimum time trajectory subject to the torque constraints. These methods, however, are unable to handle the optimal tracking problem and cannot limit the joint velocity, acceleration and jerk exactly. Dynamic programming (DP) is an algorithmic approach that is widely used for solving the optimization problem. DP has few restrictions on the performance index to be minimized, and it makes possible to take into account the various constraints. Since the dimension of the dynamic equations in the formulated problem is two, the problem of curse of dimensionality [15] can be avoided. The optimality of DP was proven by the principle of optimality [15]. From these reasons, it is indispensable to introduce DP into our problem. Vukobratovic *et al.* [12] applied the method of DP for solving the problems of energy optimization of the robot trajectory. Shin *et al.* [13] and Singh *et al.* [14] verified the usefulness of DP in solving the optimal trajectory planning problem that have the specified path.

A. Discretization and Constraints Identification

To apply DP for solving the problem, the formulation should be restated in discrete form. Let the time interval $t_f t_o$ be divided into N small subintervals $\Delta t = (t_f - t_o)/N$. Then the dynamic equations of a robot manipulator in (16a) and (16b) are discretized as

$$x_1(k+1) \simeq x_1(k) + \Delta t\, x_2(k) \tag{22a}$$

$$x_2(k+1) \simeq x_2(k) + \Delta t \frac{b(x_1(k))x_2(k)^2 + c(x_1(k), x_2(k))}{a(x_1(k))} + \Delta t \sum_{i=1}^{n} d_i(x_1(k))u_i(k) \tag{22b}$$

and the performance index in (19) is also discretized as

$$\boldsymbol{J} = \Delta t \sum_{k=0}^{N-1} (x_1(k) - \widetilde{x_1}(k))^2 + (x_2(k) - \widetilde{x_2}(k))^2 \tag{23}$$

where $x_1(k)$, $x_2(k)$, and $u_i(k)$ are the values at the time instant $t_k = t_o + k\Delta t$. Next, the constraints for the conveyor tracking are taken into account in discrete form. The joint torques generated at the time instant t_k must be limited by

$$u_{i,\min}(k) \le u_i(k) \le u_i(k), \qquad i = 1, \cdots, n \tag{5'}$$

and these inequalities are discrete form of (5). With the approximations

$$\ddot{\boldsymbol{q}}(k) \simeq (\dot{\boldsymbol{q}}(k+1) - \dot{\boldsymbol{q}}(k))/\Delta t \tag{24}$$

and

$$\boldsymbol{q}^{(3)}(k) \simeq (\dot{\boldsymbol{q}}(k+2) - 2\dot{\boldsymbol{q}}(k+1) + \dot{\boldsymbol{q}}(k))/(\Delta t)^2 \tag{25}$$

the constraints on the joint acceleration and jerk in (7) and (8) are transformed into those on the joint velocity [11]. From the inverse kinematic relation in (12), the constraints given in (6)–(8) are reformed respectively as

$$V_{i,\min} \le [d\boldsymbol{f}(x_1(k))/dx_1(k)]_i x_2(k) \le V_{i,\max} \tag{6'}$$

$$\begin{aligned} \dot{q}_i(k+1) - \Delta t A_{i,\max} &\le [d\boldsymbol{f}(x_1(k))/dx_1(k)]_i x_2(k) \\ &\le \dot{q}_i(k+1) - \Delta t A_{i,\min} \end{aligned} \tag{7'}$$

$$\begin{aligned} \dot{q}_i(k+1) - \Delta t \ddot{q}_i(k+1) + (\Delta t)^2 J_{i,\min} \\ \le [d\boldsymbol{f}(x_1(k))/dx_1(k)]_i x_2(k) \\ \le \dot{q}_i(k+1) - \Delta t \ddot{q}_i(k+1) + (\Delta t)^2 J_{i,\max} \end{aligned} \tag{8'}$$

where $i = 1, \cdots, n$. Combining (6')–(8'), we can obtain the constraints on the states at the time instant t_k, that is

$$\delta_i^-(k+1) \le [d\boldsymbol{f}(x_1(k))/dx_1(k)]_i x_2(k) \le \delta_i^+(k+1), \qquad i = 1, \cdots, n \tag{26}$$

where for the ith joint, $\delta_i^-(k+1)$ is the maximum value of three lower bounds, and $\delta_i^+(k+1)$ is the minimum value of three upper bounds. Both $\delta_i^-(k+1)$ and $\delta_i^+(k+1)$ depend on the joint velocity and acceleration at the time instant t_{k+1}. Manipulation of these inequalities gives

$$SC_{\min}(x_1(k), k+1) \le x_2(k) \le SC_{\max}(x_1(k), k+1) \tag{27}$$

where (28) and (29) result, as shown on the bottom of the page.

Together with $V_{i,\max}$, $V_{i,\min}$, $A_{i,\max}$, $A_{i,\min}$, $J_{i,\max}$ and $J_{i,\min}$ of joints, the state $x_1(k)$ and the joint velocity and acceleration at the time instant t_{k+1} put limits on the bound of the state $x_2(k)$. The constraints on the joint velocity, acceleration, and jerk that guarantee the smooth tracking motion of a robot, are now identified with those on the hand state. Optimization procedure using DP can accommodate the constraints in (5') and (27) plainly.

B. Dynamic Programming for Conveyor Tracking

The DP algorithm for solving the conveyor tracking problem may now be stated. The algorithm is named DPCT, an abbreviation of dynamic programming for conveyor tracking. The DPCT developed in this paper uses an approach similar to the one developed by Shin, *et al.* [13]. Although the two approaches share the basic ideas, they differs in the solution strategy, constraints considered and the initialization of a phase plane. Optimization procedure of the DPCT is done entirely on a $x_1 - x_2$ phase plane. The phase plane is divided into rectangular grids with N_1 divisions on the x_1 axis and N_2 divisions on the x_2 axis. The initial state $(x_1(0), x_2(0))$ and the final state $(x_1(N), x_2(N))$ are stored in the points $(0,0)$ and (N_1, N_t), respectively. Since the overshoot of the hand velocity should be taken into account, the $x_2(N)$, the desired value of $x_2(k)$, is stored in N_t that is less than N_2. The minimum performance index J_{xy}^* to reach the final point (N_1, N_t) from the point (x, y) is stored in each point. Also, the time interval t_{xy} to go from $(0,0)$ to (x, y) and the pointer P_{xy} to indicate the optimal state from (x, y) are associated with each point.

With this initialization, the DPCT is achieved by starting from the final point and proceeding backward through the following steps.

[Step 1] Let the $J_{xy}^* = 0$ and the column counter $\alpha = N_1 - 1$.
[Step 2] If $\alpha = 0$, go to [step 13].
[Step 3] Otherwise, set the $x_1(k)$ to the αth value on the x_1 axis, and compute the $d\boldsymbol{f}(x_1(k))/dx_1(k)$, $d^2\boldsymbol{f}(x_1(k))/dx_1(k)^2$, $a(x_1(k))$, $b(x_1(k))$ and $d_i(x_1(k))$.
[Step 4] Set the row counter β to 0.
[Step 5] If $\beta = N_2$, go to [Step 12].
[Step 6] Otherwise, set the $x_2(k)$ to the βth value on the x_2 axis, and compute the $c(x_1(k), x_2(k))$.
[Step 7] Set the next-row counter β to 0.
[Step 8] If $\beta = N_2$, go to [Step 11].
[Step 9] Otherwise, set the $(x_1(k+1), x_2(k+1))$ to the value of a point $(\alpha + 1, \beta)$.

[Step 9a] Compute the time interval Δt to go from (α, β) to $(\alpha + 1, \gamma)$, and determine the time instant $t_{\alpha\beta} = t_{\alpha+1,\beta} - \Delta t$, where $t_{N_1-1,\beta} = t_f - \Delta t$.
[Step 9b] Determine the $SC_{\min}(x_1(k).k+1)$ and $SC_{\max}(x_1(k), k+1)$. If $x_2(k)$ violates its constraint in (27), go to [Step 10].
[Step 9c] Compute the torques $u_i(k)$, which lead the motion of a robot manipulator from the state $(x_1(k), x_2(k))$ into the state $(x_1(k+1), x_2(k+1))$. If any of these computed torques violates its constraints, go to [Step 10].
[Step 9d] Compute the part state, $\widetilde{x_1}(k)$ and $\widetilde{x_2}(k)$, by plugging the $t_{\alpha\beta}$ into (17) and (18), respectively. From

$$SC_{\max}(x_1(k), k+1) = \min_{i=1,\cdots,n} \begin{cases} \dfrac{\delta_i^+(k+1)}{[d\boldsymbol{f}(x_1(k))/dx_1(k)]_i} & \text{when } [d\boldsymbol{f}(x_1(k))/dx_1(k)]_i > 0 \\ \dfrac{\delta_i^-(k+1)}{[d\boldsymbol{f}(x_1(k))/dx_1(k)]_i} & \text{when } [d\boldsymbol{f}(x_1(k))/dx_1(k)]_i < 0 \end{cases} \tag{28}$$

$$SC_{\min}(x_1(k), k+1) = \max_{i=1,\cdots,n} \begin{cases} \dfrac{\delta_i^-(k+1)}{[d\boldsymbol{f}(x_1(k))/dx_1(k)]_i} & \text{when } [d\boldsymbol{f}(x_1(k))/dx_1(k)]_i > 0 \\ \dfrac{\delta_i^+(k+1)}{[d\boldsymbol{f}(x_1(k))/dx_1(k)]_i} & \text{when } [d\boldsymbol{f}(x_1(k))/dx_1(k)]_i < 0 \end{cases} \tag{29}$$

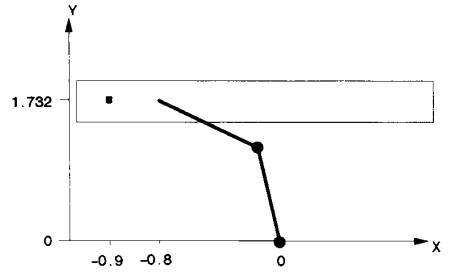


Fig. 2. Simulation model.

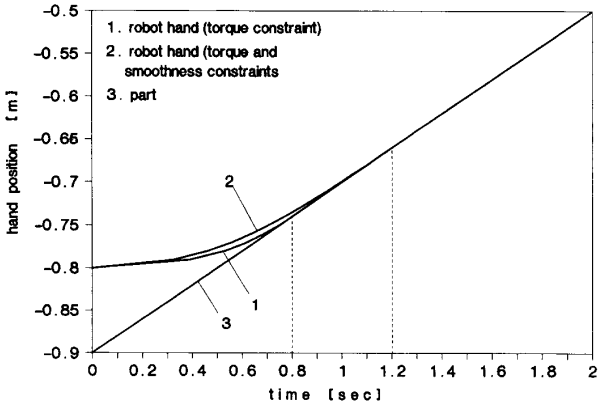


Fig. 3. Position trajectories for conveyor tracking.

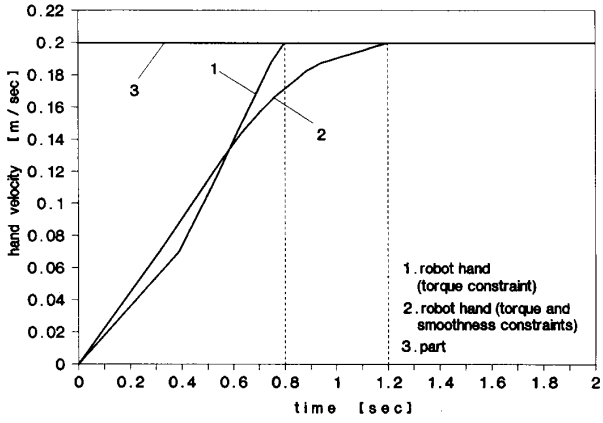


Fig. 4. Velocity trajectories for conveyor tracking.

the part state, determine the incremental performance index ΔJ to go to $(\alpha+1, \gamma)$ from (α,β) as

$$\Delta J = \Delta t\big((x_1(k) - \widetilde{x_1}(k))^2 + (x_2(k) - \widetilde{x_2}(k)^2)\big) \tag{30}$$

The minimum performance index $J^*_{\alpha\beta}$ is then

$$J^*_{\alpha,\beta} = \min_{\gamma=0,\cdots,N_2} \big(\Delta J + J^*_{\alpha+1,\gamma}\big). \tag{31}$$

Set the pointer $P_{\alpha\beta}$ to β that produces the $J^*_{\alpha\beta}$.

[Step 10] Increment the next-row counter γ and go to [Step 8].

[Step 11] Increment the row counter β and go to [Step 5].

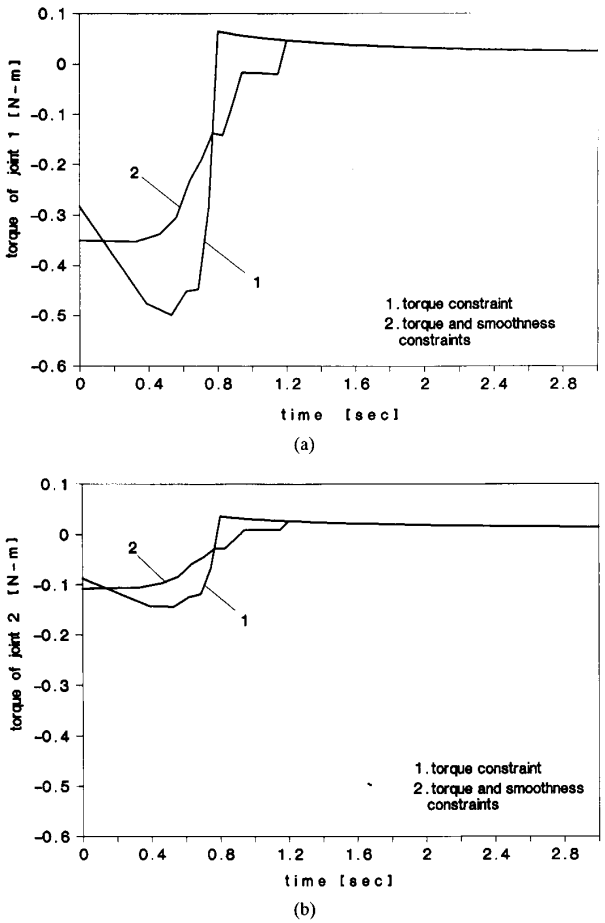


Fig. 5. Torque profiles for conveyor tracking. (a) Joint 1 (b) Joint 2.

[Step 12] Decrement the column counter α and go to [Step 2].

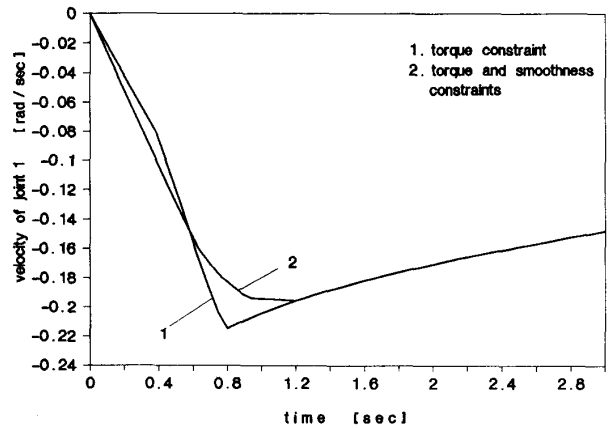
[Step 13] March forward from the initial point (0,0), by following the pointer $P_{\alpha\beta}$, to obtain the optimal phase plot.

V. SIMULATION STUDY

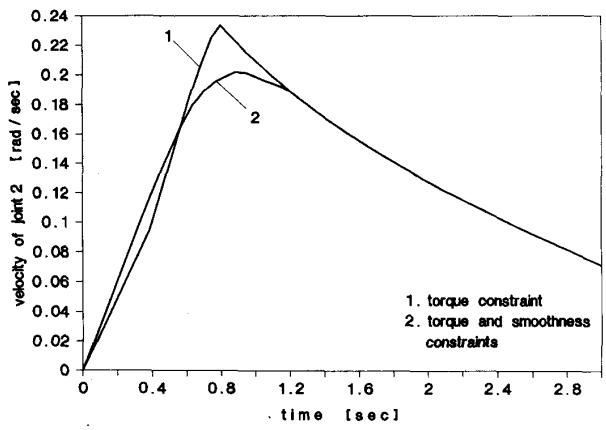
The described procedure is applied to a robotic workcell as shown in Fig. 2, which consists of a conveyor belt system and a manipulator. The manipulator has two links with a mass of 1.0 kg, where the masses are assumed to be concentrated at the middle of each link. The length of each link is 1.0 m. This manipulator has two rotary joints and the axes of rotation are both directed along the z axis, thus the manipulator only generates movement on the x–y plane. Gravity is presumed to act in negative z direction. The algorithm can be easily applied to nonredundant manipulators with more degrees of freedom. The conveyor belt moves in x direction at a constant velocity $\boldsymbol{\nu_b} = [0.2\ 0.0]^T$ m/s. The part is stationary with respect to the moving conveyor belt, and it passes the position $\boldsymbol{x_p}(t_o) = [-0.900\ 1.732]^T$ m at the initial time. On the other hand, at the initial time, the manipulator is at rest on the position $\boldsymbol{x_h}(t_o) = [-0.800\ 1.732]^T$ m and starts to move in x direction for tracking the part. The torque and smoothness constraints of the manipulator are assumed to be given as

$$-0.50 \le u_1(t) \le 0.50, \quad -0.20 \le u_2(t) \le 0.20 \text{ N-m} \tag{32}$$

$$-0.20 \le \dot{q}_1(t) \le 0.20, \quad -0.21 \le \dot{q}_2(t) \le 0.21 \text{ rad/s} \tag{33}$$

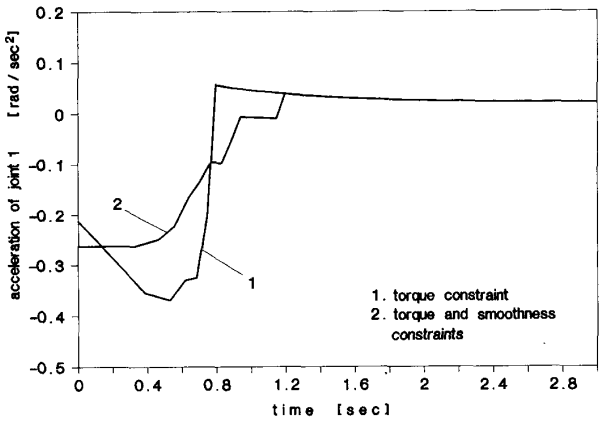


(a)

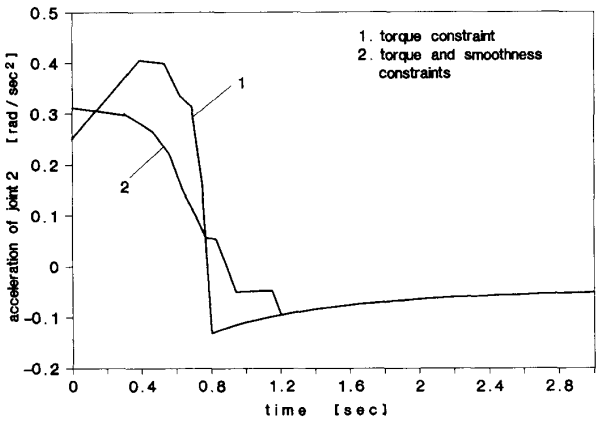


(b)

Fig. 6. Joint velocity profiles for conveyor tracking. (a) joint 1 (b) joint 2.



(a)



(b)

Fig. 7. Joint acceleration profiles for conveyor tracking. (a) joint 1 (b) joint 2.

$$-0.29 \le \ddot{q}_1(t) \le 0.29, \quad -0.32 \le \ddot{q}_2(t) \le 0.32 \text{ rad/s}^2 \quad (34)$$

$$-0.96 \le q_1^{(3)}(t) \le 0.96, \quad -0.96 \le q_2^{(3)}(t) \le 0.96 \text{ rad/s}^3. \quad (35)$$

The DPCT algorithm was implemented in the C programming language running under VMS operating system on a micro-VAX 2.

Trajectories for conveyor tracking are generated and compared for the cases of 1) torque constraints in (32) and 2) both torque and smoothness constraints in (32)–(35). $N_1 \times N_2$ is set at 50×100. Fig. 3 shows the position of the robot hand, and Fig. 4 shows the velocity of the robot hand when it tracks the part on the conveyor belt system. In all two cases, the tracking error in the steady state is eliminated completely. The settling time, defined in this simulation as the instant when the tracking error remains zero, is 0.80(s) for case 1) and 1.20 s for case 1).

Fig. 5 shows the torque profiles generated in each joint. In order to go into the transient state, large torques are required in the transient state. For case 1), the maximum torques are -0.499 N-m for joint 1 and -0.140 N-m for joint 2, and they are all limited within theirs bounds in (32). When only the torque constraints are applied, torques change suddenly in the instant when the robot reaches the steady state, which may result in the undesirable motion of a robot. Introducing the smoothness constraints together with the torque constraints, induces the smooth torque profiles. Fig. 6 shows the joint velocity profiles, and Fig. 7 shows the joint acceleration profiles. When no smoothness constraints are imposed, the steady state is reached with sudden changes in velocities, i.e., smooth velocity profiles cannot be obtained with torque constraints only. Applying the jerk constraints prevents the accelerations from their sudden change.

VI. CONCLUSION

The robot motion for conveyor tracking in a robotic workcell has been analyzed and planned so far. In the presented motion analysis and planning, the robot dynamics was included, and the speed of the conveyor belt and the locations of the part and the robot were considered. The joint torque limit, together with the joint velocity, acceleration and jerk limits of the robot, is also taken into account. A conveyor tracking problem was defined with some assumptions, and was converted to an optimal tracking problem. A solution that minimizes the specified performance index subject to the various constraints was then obtained using a dynamic programming approach. In the simulation study, the robot trajectories without the steady state tracking error were planned. Large joint torques were generated in the transient state, and the maximum values were limited by introducing the torque constraints. The joint torque and acceleration changed suddenly in the instant when the robot go into the steady state, which was prevented by imposing the smoothness constraints.

In order to maximize the task efficiency of conveyor-tracking applications, it is desirable to design the conveyor belt system

coupled with a robot manipulator considering the arm dynamics and constraints. Our results can be used for this purpose. In the future, we will try to solve the conveyor tracking problem with less assumptions.

REFERENCES

[1] W. E. Wilhelm, "Conveyor tracking," in *International Encyclopedia of Robotics: Applications and Automation*, R. C. Dorf, Ed. New York: Wiley, 1988, pp. 283–298.

[2] S. W. Holland, L. Rossal and M. R. Ward, "CONSIGHT-1: A vision controlled robot system for transferring parts from belt conveyors," in *Computer Vision and Sensor-Based Robots*, G. G. Dodd and L. Rossal, Eds. New York: Plenum, 1979.

[3] X. J. Mo and L. Z. Liu, "A robot system with vision, touch and slide sensors for the grip onto a moving conveyor belt," in *Proc. 15th Int. Symp. Industrial Robots*, Tokyo, Japan, Sept. 1985, pp. 129–136.

[4] W. Patzelt, "A robot position control algorithm for the grip onto an accelerated conveyor belt," in *Proc. 12th Int. Symp. Industrial Robots*, Paris, France, June 1982, pp. 391–399.

[5] J. Y. S. Luh, "Conventional controller design for industrial robots—A tutorial," *IEEE Trans. Syst., Man, Cybern.*, vol. SMC-13, pp. 298–316, May/June 1983.

[6] R. P. Paul, "Manipulator cartesian path control," *IEEE Trans. Syst., Man, Cybern.*, vol. SMC-9, pp. 702–711, Nov. 1979.

[7] R. H. Taylor, "Planning and execution of straight line manipulator trajectories," *IBM J. Res. Develop.*, vol. 23, no. 4, pp. 424–436, July 1979.

[8] H. P. Geering, L. Guzzella, S. A. R. Hepner and C. H. Onder, "Time-optimal motions of robots in assembly tasks," *IEEE Trans. Automatic Contr.*, vol. AC-31, pp. 512–518, June 1986.

[9] K. G. Shin and N. D. Mckay, "Minimum-time control of robotic manipulators with geometric path constraints," *IEEE Trans. Automatic Contr.*, vol. AC-30, pp. 531–541, June 1985.

[10] J. E. Bobrow, S. Dubosky and J. S. Gibson, "Time-optimal control of robotic manipulators along specified path," *Int. J. Robotics Res.*, vol. 4, no. 3, pp. 3–17, Fall 1985.

[11] B. H. Lee, "Constraint identification in time-varying obstacle avoidance for robot manipulators," *IEEE Trans. Syst., Man, Cybern.*, vol. SMC-19, pp. 140–143, Jan./Feb. 1989.

[12] M. Vukobratovic and M. Kirkanski, "A method for optimal synthesis of manipulation robot trajectories," *ASME J. Dyn. Syst., Meas., Contr.*, vol. 104, no. 2, pp. 188–193, June 1982.

[13] K. G. Shin and N. D. Mckay, "A dynamic programming approach to trajectory planning of robotic manipulators," *IEEE Trans. Automatic Contr.*, vol. AC-31, pp. 491–500, June 1986.

[14] S. Singh and M. C. Leu, "Optimal trajectory generation for manipulators using dynamic programming," ASME *J. Dyn. Syst., Meas., Contr.*, vol. 109, no. 2, pp. 88–96, June 1987.

[15] D. E. Kirk, *Optimal Control Theory*. Englewood Cliffs, NJ: Prentice-Hall, 1970.

[16] R. E. Bellman and S. E. Dreyfus, *Applied Dynamic Programming*. Princeton, NJ: Princeton Univ. Press, 1970.

[17] M. Brady, J. M. Hollerbach, T. L. Johnson, T. L. Perez and M. T. *Johnson, Robot Motion: Planning and Control*. Cambridge, MA: MIT Press, 1982.

Toward Super-Real-Time Simulation of Robotic Mechanisms Using a Parallel Integration Method

Scott McMillan, David E. Orin, and P. Sadayappan

***Abstract*—The desirability of real-time and super-real-time (planning seconds of trajectory in milliseconds) dynamic simulation has been shown in a number of applications such as in earth-based teleoperation of remote robotic systems in space, as well as in predictive control of multilegged vehicles, used to ensure the safety and stability along a planned trajectory. The results of research performed in computational robot dynamics to achieve real-time simulation of a manipulator on a general-purpose vector/parallel computer are presented. After effective vectorization of the complex dynamics equations required in simulation, a coarse-grain parallel block predictor-corrector (BPC) method for performing the motion integration was realized on multiple processors to exploit a form of temporal parallelism. Results on a CRAY Y-MP8/864 show that effective use of vectorization and parallelization yields an order of magnitude speedup resulting in a computational rate 50 times faster than real-time for end-effector position errors on the order of a micron. This translates to real-time performance on a less powerful parallel computing system.**

I. INTRODUCTION

Dynamic simulation is an important tool that has been limited to off-line applications such as the evaluation of mechanical and control designs for robotic systems. With its use, basic approaches to mechanical design may be compared while avoiding construction of costly prototypes. It also allows control engineers to develop the basic structure of the controller and set initial values for the gains that stabilize the system while avoiding the ill effects of instabilities on actual hardware.

The desirability of real-time simulation for on-line control has been shown in a number of applications such as in earth-based teleoperation of remote robotic systems in space [1]. Because of the transmission time delays involved in satellite communications, direct control by a human operator working with a delayed image of the robotic system is difficult. Performance is greatly improved when a display of the simulated system is also made available to the operator, on-line, with no time delay. While many of the current teleoperated systems are slow enough so that a kinematic model is sufficient, dynamic simulations may be required when force control or faster trajectories are desired. Beyond the real-time goal, *super*-real-time simulation is also desired in some applications such as in advanced control schemes where trajectory planning is used, and seconds of motion trajectory need to be simulated in milliseconds [2]. This is useful in multilegged vehicles [3] when prediction of the action of the present control is used to ensure safety and stability along a desired trajectory.

The major obstacle to achieving the required computational rates, however, is the complexity of the dynamic equations to be simulated. Because of this, dynamic simulation is usually several orders of magnitude slower than real-time, and this has often limited its utility. The problem is compounded by increases in the structural and task complexity found in systems now being considered. These systems

Manuscript received July 14, 1990; revised May 24, 1991. This work was supported in part by a grant from CRAY Research and in part by The Ohio State University.

S. McMillan and D. E. Orin are with the Department of Electrical Engineering, The Ohio State University, Columbus, OH 43210.

P. Sadayappan is with the Department of Computer and Information Science, The Ohio State University, Columbus, OH 43210.

IEEE Log Number 9104098.