L. G. Van Willigenburg
Department of Agricultural Engineering and Physics
Wageningen University
Bomenweg 4, 6703 HD, Wageningen
The Netherlands
E-mail: gerard.vanwilligenburg@user.aenf.wau.nl

Computation and Implementation of Digital Time-Optimal Feedback Controllers for an Industrial X-Y Robot subjected to Path, Torque, and Velocity Constraints

Abstract

For an industrial X-Y robot, in which the links are subjected to torque and velocity constraints, the time-optimal control problem is solved where the robot motion is constrained to follow an arbitrary path. A numerical procedure to compute the solution is presented and demonstrated. The solution consists of a continuous-time state trajectory and open-loop control. Because the X-Y robot is controlled by a digital computer, a recently developed numerical procedure to compute optimal tracking digital controllers is applied to the solution to arrive at an implementable digital time-optimal feedback controller that accounts for modeling errors as well. Experimental results obtained after implementation of the digital time-optimal feedback controllers are presented for two paths. The robot dynamics include both viscous and Coulomb friction. The extension of the solution and numerical procedures to general rigid manipulators, including both viscous and Coulomb friction, is straightforward.

1. Introduction

Manufacturing operations performed by robots (e.g., spraying, cutting, or welding) are specified through a desired robot motion in space. However, there is freedom in how this motion should be executed in time. Because generally one wants the robot to operate as fast as possible, the problem of executing a motion specified in space, called a path, in minimum time is of vital practical importance. Furthermore, the solution of the problem may be used to select paths with equal starting and end points, according to their minimum traveling time, in an attempt to solve another time-optimal control problem. This problem considers the time-optimal motion of a robot subjected to actuation torque constraints (Ailon and Langholtz 1985; Geering et al. 1986; Sontag and Sussman 1986; Van Willigenburg 1991)—and possibly gripper, payload, and obstacle constraints (Shiller and Dubowsky 1989)—from a specified initial configuration to a specified final configuration (i.e., a time-optimal point to point motion).

The time-optimal control problem, in which robot motion is constrained to follow an arbitrary path, has already been solved by several authors (Shiller and Dubowsky 1989; Bobrow et al. 1985; Shin and McKay 1985). However, these authors consider only one type of actuator constraint—i.e., the limits on the actuation torques of the links—while neglecting the velocity limits of the links, which play a major role in practice. These limits are caused by another actuator constraint and possibly by the mechanical constraints of the robot mechanism. A detailed description of a numerical algorithm to compute the solution and experimental results after implementation was not presented by any of the authors. Except for Shin and McKay (1985), who consider viscous friction, friction is not included in the robot dynamics, although both viscous and Coulomb friction play a significant role in practice (Craig 1986).

We present a solution to the time-optimal control problem for an industrial X-Y robot in which the motion is constrained to follow an arbitrary path, and both the actuation torque and velocity constraints of the links are considered. The robot dynamics include both viscous and Coulomb friction. A numerical procedure to compute the

The International Journal of Robotics Research,
Vol. 12, No. 5, October 1993, pp. 420–433,

solution will be presented along with numerical examples involving two paths. Although not treated in this article, the extension of the solution and the numerical procedure to general rigid manipulators, where both Coulomb and viscous friction may be included in the manipulator dynamics, is straightforward.

The solution to the problem in which the robot has to travel a prescribed path in minimum time consists of a continuous-time state trajectory and a continuous-time open-loop control. Because robots are controlled by computers in practice, we are unable to generate continuous-time controls. Therefore, we apply a recently developed numerical procedure to compute digital optimal tracking controllers (Van Willigenburg 1991) to arrive at an implementable digital time-optimal feedback controller that also accounts for modeling errors.

2. Actuators and Constraints

The industrial X-Y robot that we consider is actuated by DC motors. The dynamics of a DC motor can be represented by

$$U = L dI/dt + RI + K\omega, \tag{1a}$$

$$T = KI, \tag{1b}$$

where

L = the induction of the motor (H),
R = the electrical resistance of the motor (Ohm),
I = the motor current (A),
U = the voltage applied to the motor (V),
ω = the angular speed of the motor (rad/s),
T = torque generated by the motor (N · m/rad),
K = Voltage constant of the motor (Vs/rad).

From both a practical and a modeling viewpoint, it will be convenient to choose the motor currents to be the control variables of the robot. The capabilities of a DC motor are mainly limited by the heat generation and dissipation characteristics. Heat generation is represented by the second term in (1a) and so is proportional to the motor current. Conventional DC motor controllers are built around a motor current controller, which is used to limit the motor current to prevent overheating. If we consider the current controller to be ideal, which is a reasonable assumption (Van Willigenburg 1991), and send our actual control signals to the inputs of the current controllers, we may consider the motor currents to be the control variables. A practical advantage of this approach is that we need only the motor current controller and not other parts of conventional DC motor controllers.

Very often, dynamic models of robots consider forces and/or torques applied to the robot links to be the control variables. From (1) it can be seen that the motor current is proportional to the torque. If we assume the transmission from the DC motor to the links to be ideal, the force applied to each robot link is proportional to the motor current. The assumption that the motor currents are the control variables in this case does not affect the structure of the robot dynamics. In practice the transmission suffers from gear cogging, backlash, and bearing nonlinearities. In our case these constitute modeling errors that will be compensated for by the digital feedback controller designed in Section 7.

We will adopt the assumptions mentioned earlier and consider the DC motor currents to be the control variables. One actuator constraint consists of the limitation of the motor current to prevent overheating,

$$-I_{\max} \leq I \leq I_{\max}, \tag{2}$$

where $I_{\max}$ represents the maximum value of the motor current where overheating will not occur. In practice, to control the motor current, the voltage supplied to the DC motor is rapidly switched on and off, which may be considered as varying the voltage supplied to the motor. Therefore, we have

$$-U_{\max} \leq U \leq U_{\max}, \tag{3}$$

where $U_{\max}$ represents the voltage of the switched bipolar voltage supply. In our case, and in fact very often, we may neglect the first term in equation (1). Doing so, from (1) and (3) we observe that the largest interval to which ω may belong that still allows the motor current to reach the complete interval given by (2) is given by,

$$\omega \in [-\omega_{\max}, \omega_{\max}] \tag{4a}$$

where

$$\omega_{\max} = (U_{\max} - RI_{\max})/K \tag{4b}$$

We will use constraint (4), although it is conservative, as this allows for a constant bound (2) for the motor currents (i.e., the control variables). If the bound (4) is considered to be a serious drawback, increasing the value of the voltage of the switched voltage supply $U_{\max}$ will increase the bound (4) without violating other DC motor constraints. Because we assume the X-Y robot to be rigid and the transmission to be ideal, the actuator constraint (4) can be transformed to a constraint on the link velocity, which in this case is proportional to ω. Thus in our case the velocity limits on the individual links are caused by an actuator constraint. Velocity limits on the individual links may also result from mechanical constraints on the robot mechanism.

3. Dynamic Model of the X-Y Robot

Very often friction effects are neglected in robot dynamics (Ailon and Langholtz 1985; Geering et al. 1986; Bobrow

et al. 1985; Shiller and Dubowsky 1989). Experiments with our X-Y robot have demonstrated that friction effects play a significant role in the dynamic behavior (Van Willigenburg 1991). In our robot model, friction is represented by both viscous and Coulomb friction. Using this together with the other assumption, we arrive at the following robot model:

$$\ddot{x}_p = -v_x \dot{x}_p - c_x \operatorname{sign}(\dot{x}_p) + b_x u_x, \tag{5a}$$

$$\ddot{y}_p = -v_y \dot{y}_p - c_y \operatorname{sign}(\dot{y}_p) + b_y u_y, \tag{5b}$$

where x_p represents the translation of the x-link with respect to a reference position, y_p represents the translation of the y-link with respect to a reference position, v_x and v_y are the viscous friction coefficients, c_x and c_y the Coulomb friction coefficients, and b_x and b_y the sensitivity coefficients to the control variables u_x and u_y, which are equal to the motor currents. Both b_x and b_y depend on the mass of the link and the payload, which we assume to be known. We rewrite the dynamics (5) in state-space form as

$$\begin{bmatrix} \dot{x}_p \\ \dot{y}_p \\ \ddot{x}_p \\ \ddot{y}_p \end{bmatrix} = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & -v_x & 0 \\ 0 & 0 & -v_y & 0 \end{bmatrix} \begin{bmatrix} x_p \\ y_p \\ \dot{x}_p \\ \dot{y}_p \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ b_x & 0 \\ 0 & b_y \end{bmatrix} \begin{bmatrix} u_x \\ u_y \end{bmatrix} - \begin{bmatrix} 0 \\ 0 \\ c_x \operatorname{sign}(\dot{x}_p) \\ c_y \operatorname{sign}(\dot{y}_p) \end{bmatrix}. \tag{6}$$

The actuator constraint (2) in terms of (5) and (6) becomes a constraint on the control

$$|u_x| \le u_{\max}, \tag{7a}$$

$$|u_y| \le u_{\max}. \tag{7b}$$

where

$$u_{\max} = I_{\max}. \tag{7c}$$

Given the assumption of an ideal transmission, we have

$$\dot{x}_p = c_t \omega, \tag{8}$$

where c_t is a positive constant determined by the transmission. Together with (4) we obtain the following state constraint:

$$|\dot{x}_p| \le s_{\max}, \tag{9a}$$

where

$$s_{\max} = \omega_{\max}/c_t. \tag{9b}$$

The same constraint holds for $\dot{y}_p$ the speed of the y-link:

$$|\dot{y}_p| \le s_{\max}. \tag{9c}$$

The actuator constraint (7) limits the force applied to each link. This limits both the link speeds and accelerations, as will be demonstrated in the next section. The state constraint (9) directly limits the link speeds.

4. The Time-Optimal Control Problem

We consider the problem where the X-Y robot described by (6), (7), and (9) has to travel a prescribed path in minimum time. The path prescribes the evolution of half the number of state variables (i.e., x_p and y_p, the link positions); however, it does so not as a function of time, but as a function of a certain parameter λ:

$$x_p = f(\lambda), \; y_p = g(\lambda), \qquad 0 \le \lambda \le \lambda_{\max}. \tag{10}$$

In other words the path (10) prescribes the robot motion in space. The functions f and g are continuous functions of λ with continuous first derivatives. In practice the functions f and g in (10) are seldom directly available. Usually the path is specified as a finite number of coordinate pairs (x_p, y_p) that have to be passed in a given order. In Section 6 we will use cubic splines to interpolate the x_p and y_p coordinates. Thus both f and g will be a piecewise polynomial of order 4, and therefore a piecewise analytic function, that possesses a continuous first derivative (De Boor 1978).

Given the dynamics (6) and the constraints (7), (9), and (10), the time-optimal control problem comes down to determining

$$\lambda(t) \qquad 0 \le t \le t_{\max} \tag{11a}$$

in (10) as a nondecreasing continuous function of time—i.e.,

$$\dot{\lambda}(t) \ge 0 \qquad 0 \le t \le t_{\max}, \tag{11b}$$

where

$$\lambda(0) = 0, \tag{11c}$$

$$\lambda(t_{\max}) = \lambda_{\max}, \tag{11d}$$

such that $t_{\max}$ is minimal. Once (11) is known, the time-optimal trajectory is given by (10); i.e.,

$$x_p = f(\lambda(t)) \qquad 0 \le t \le t_{\max}, \tag{12a}$$

$$y_p = g(\lambda(t)) \qquad 0 \le t \le t_{\max}. \tag{12b}$$

Note that when $\dot{\lambda}(t) = 0$, all links stand still. At these points we may split the path, and therefore the problem, in two parts by demanding the robot to stand still at the end of the first path, as well as at the start of the second. According to (6) and (12), the latter is done by prescribing $\dot{\lambda} = 0$. Therefore, except for possibly the initial and final time, we have $\dot{\lambda} > 0$—i.e.,

$$\dot{\lambda}(t) > 0, \qquad 0 < t < t_{\max}. \tag{13}$$

Because $\lambda(t)$ in (12a) is a nondecreasing continuous function of time, minimizing $t_{\max}$ is equivalent to maximizing $d\lambda/dt$ at all times t, $0 \le t \le t_{\max}$. The solution to the time-optimal control problem will be based on the conversion of the constraints (7), (9), and (10) into constraints on $d\lambda/dt$ and $d^2\lambda/dt^2$ and the conversion of the

dynamics (6) into dynamics concerning $\lambda(t)$. At this stage we adopt the following notation:

$$f_1 = df/d\lambda, \quad f_2 = d^2f/d\lambda^2,$$
$$g_1 = dg/d\lambda, \quad g_2 = d^2g/d\lambda^2,$$
$$\dot{\lambda} = d\lambda/dt, \quad \ddot{\lambda} = d^2\lambda/dt^2. \tag{14}$$

The quantity $\dot{\lambda}$ is referred to as the *path velocity*, because it determines the speed by which the path is traveled in time. Accordingly, the quantity $\ddot{\lambda}$ is referred to as the path acceleration. From (12) and (14) we establish the following facts, determined by well-known rules for differentiation:

$$\dot{x}_p = f_1\dot{\lambda}, \tag{15a}$$
$$\ddot{x}_p = f_1\ddot{\lambda} + f_2\dot{\lambda}^2, \tag{15b}$$
$$\dot{y}_p = g_1\dot{\lambda}, \tag{15c}$$
$$\ddot{y}_p = g_1\ddot{\lambda} + g_2\dot{\lambda}^2. \tag{15d}$$

From (6) we see that a state trajectory determined by $x_p(t)$, $y_p(t)$, $t_0 \le t \le t_{\max}$, can only be realized if both functions are continuous and have continuous first derivatives. Now from (15a) and (15c), given the properties of f, g, and $\lambda(t)$, we observe that this is so. Because of the constraints (7) and (9), not all state trajectories possessing these properties can actually be realized. We will show that by adjusting $\lambda(t)$ we can always satisfy the constraints (7), (9), and (10).

From (15) and (5) we obtain

$$f_1\ddot{\lambda} + f_2\dot{\lambda}^2 = -v_x f_1\dot{\lambda} - c_x \operatorname{sign}(f_1\dot{\lambda}) + b_x u_x, \quad 0 \le t \le t_{\max}, \tag{16a}$$
$$g_1\ddot{\lambda} + g_2\dot{\lambda}^2 = -v_y g_1\dot{\lambda} - c_y \operatorname{sign}(g_1\dot{\lambda}) + b_y u_y, \quad 0 \le t \le t_{\max}. \tag{16b}$$

Assuming f_1 and g_1 everywhere are nonzero, from (16) and (13) we obtain

$$\ddot{\lambda} = -f_2\dot{\lambda}^2/f_1 - v_x\dot{\lambda} - c_x/|f_1| + b_x u_x/f_1, \quad 0 < t < t_{\max}, \tag{17a}$$
$$\ddot{\lambda} = -g_2\dot{\lambda}^2/g_1 - v_y\dot{\lambda} - c_y/|g_1| + b_y u_y/g_1, \quad 0 < t < t_{\max}. \tag{17b}$$

The constraint (9) using (15a) and (15c) may be written as,

$$\dot{\lambda} \le s_{\max}/|f_1|, \quad 0 \le t \le t_{\max}, \tag{18a}$$
$$\dot{\lambda} \le s_{\max}/|g_1|, \quad 0 \le t \le t_{\max}. \tag{18b}$$

Because f_1, f_2, g_1, and g_2 are functions of only λ, equation (17) constitutes two differential equations in λ of which the evolution should be the same. The evolution is uniquely determined by $\lambda(0)$ and $\dot{\lambda}(0)$ and the values of the control variables $u_x(t)$ and $u_y(t)$, $0 \le t \le t_{\max}$. Because the evolution of both differential equations should be the same, *only one of the control variables can be chosen independently; the other (or others, for robots with more than two links) has to be adjusted to stay on the path.*

Now we are in a position to restate the time-optimal control problem.

4.1. Problem Formulation

The time-optimal control problem is to find the controls $u_x(t)$, $u_y(t)$, $0 \le t \le t_{\max}$ bounded by (7) such that the evolution of (17a) and (17b) is the same, while the path velocity $\dot{\lambda}$, bounded by (18) is maximized at all times t, given the boundary conditions $\lambda(t_0)\dot{\lambda}(t_0)$, $\lambda(t_{\max})$, and $\dot{\lambda}(t_{\max})$.

We will first examine the influence of the constraint (7) on the problem and then examine the influence of the constraint (18). The constraint (7) restricts the admissible values of the path acceleration $\ddot{\lambda}(t)$ in (17):

$$\alpha_x(\lambda,\dot{\lambda}) \le \ddot{\lambda} \le \beta_x(\lambda,\dot{\lambda}), \quad 0 \le t \le t_{\max}, \tag{19a}$$
$$\alpha_y(\lambda,\dot{\lambda}) \le \ddot{\lambda} \le \beta_y(\lambda,\dot{\lambda}), \quad 0 \le t \le t_{\max}, \tag{19b}$$

where

$$\alpha_x(\lambda,\dot{\lambda}) = -f_2\dot{\lambda}^2/f_1 - v_x\dot{\lambda} - c_x/|f_1| - b_x u_{\max}/|f_1|, \tag{19c}$$
$$\beta_x(\lambda,\dot{\lambda}) = -f_2\dot{\lambda}^2/f_1 - v_x\dot{\lambda} - c_x/|f_1| + b_x u_{\max}/|f_1|, \tag{19d}$$
$$\alpha_y(\lambda,\dot{\lambda}) = -g_2\dot{\lambda}^2/g_1 - v_y\dot{\lambda} - c_y/|g_1| - b_y u_{\max}/|g_1|, \tag{19e}$$
$$\beta_y(\lambda,\dot{\lambda}) = -g_2\dot{\lambda}^2/g_1 - v_y\dot{\lambda} - c_y/|g_1| + b_y u_{\max}/|g_1|. \tag{19f}$$

For each λ, equation (19) restricts the admissible values of the path velocity $\dot{\lambda}$. For each λ the admissible values for $\dot{\lambda}$ have to be such that

$$\alpha_x(\lambda,\dot{\lambda}) - \beta_y(\lambda,\dot{\lambda}) \le 0 \wedge \alpha_y(\lambda,\dot{\lambda}) - \beta_x(\lambda,\dot{\lambda}) \le 0. \tag{20}$$

In general, if, for some λ, the largest α exceeds the smallest β, $\dot{\lambda}$ is inadmissible. Therefore (20) is equivalent to

$$\alpha_{\max}(\lambda,\dot{\lambda}) - \beta_{\min}(\lambda,\dot{\lambda}) \le 0, \tag{21a}$$

where

$$\alpha_{\max} = \mathrm{MAX}(\alpha_x,\alpha_y), \tag{21b}$$
$$\beta_{\min} = \mathrm{MIN}(\beta_x,\beta_y). \tag{21c}$$

In (21) MAX refers to the maximum operation applied to the values within the brackets and MIN to the minimum operation. In general if the robot has n links, (21) is

equivalent to $n(n-1)$ inequalities. Using (19), we may rewrite (20) as

$$p_x(\lambda, \dot{\lambda}) \leq 0 \wedge p_y(\lambda, \dot{\lambda}), \leq 0, \tag{22a}$$

where

$$p_x(\lambda, \dot{\lambda}) = -a\dot{\lambda}^2 - b\dot{\lambda} - c - d, \tag{22b}$$

$$p_y(\lambda, \dot{\lambda}) = a\dot{\lambda}^2 + b\dot{\lambda} + c - d, \tag{22c}$$

where

$$a = f_2/f_1 - g_2/g_1, \tag{22d}$$

$$b = v_x - v_y, \tag{22e}$$

$$c = c_x/|f_1| - c_y/|g_1| \tag{22f}$$

$$d = (b_x/|f_1| + b_y/|g_1|)u_{\max}. \tag{22g}$$

For each λ, equation (22) contains quadratic inequalities in $\dot{\lambda}$ that may be visualized by parabolas (Fig. 1). Now because f_1 and g_1 are continuous, the robot is always capable of following a path if it is moving slow enough. In terms of (12), this implies that each constraint in (22) must allow values of the path velocity $\dot{\lambda}$ within an interval $[0, \varepsilon)$, where ε is a small positive constant. With this in mind, we can visualize the two possibilities by which (22) may put constraints on $\dot{\lambda}$. From Figure 1 we observe that the admissible values of $\dot{\lambda}$ for each λ, may consist of either a single interval or two distinct intervals. If we draw the admissible values for $\dot{\lambda}$ against λ, two typical situations may arise. If the admissible values of $\dot{\lambda}$ for each λ consist of a single interval, the admissible region will contain no "islands of inadmissibility" (Fig. 2). Otherwise, the admissible region will contain at least one "island of inadmissibility" (Shin and McKay 1985). The border $T(\lambda)$ of the admissible region, according to (21), is determined by

$$\alpha_{\max}(\lambda, \dot{\lambda}) - \beta_{\min}(\lambda, \dot{\lambda}) = 0. \tag{23}$$

In the following, for convenience, although it is sometimes formally incorrect, we will assume the border $T(\lambda)$ to belong to the admissible region. In practice (see Section 7) we always have to choose conservative bounds for the control variables, and so this assumption is legitimate. Note that the $(\lambda, \dot{\lambda})$ plane is often referred to as the *phase plane* (Shin and McKay 1985).

In Figure 2, $T(\lambda)$ constitutes the border between the admissible and inadmissible regions caused by constraints (7) and (10). Constraints (7) and (10) also cause the bounds (19) on the path acceleration $\ddot{\lambda}$. We may represent these bounds in Figure 2 by introducing

$$\mu = d\dot{\lambda}/d\lambda. \tag{24}$$

The quantity μ represents the slope of the path velocity evolution as a function of λ. From (24) we have

$$\mu = d\dot{\lambda}/d\lambda = \frac{d\dot{\lambda}/dt}{d\lambda/dt} = \ddot{\lambda}/\dot{\lambda}. \tag{25}$$

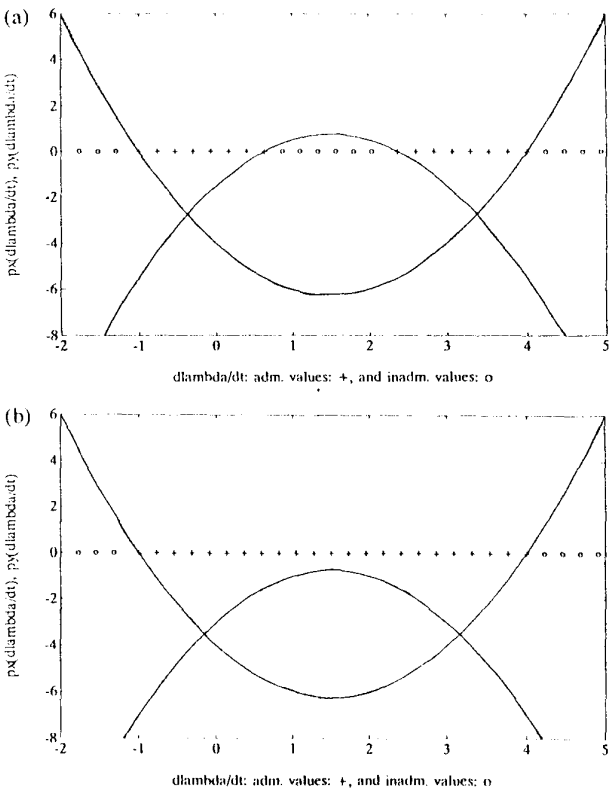

Fig. 1. A, Admissable and inadmissable values of $d\lambda/dt$ for a single lambda. B, Admissable and inadmissable values of $d\lambda/dt$ for a single lambda.

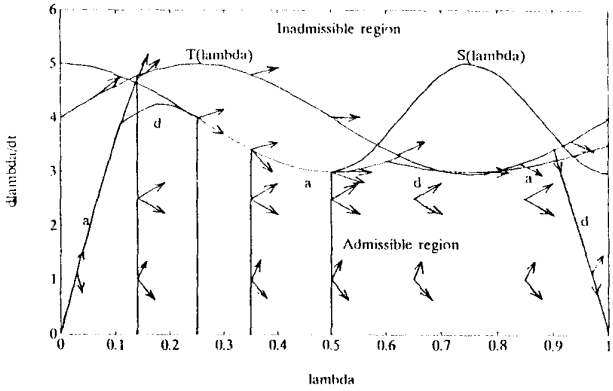

Fig. 2. Problem and solution in the phase plane.

From (25) we observe that the admissible values of μ at each point in Figure 2 are bounded by (19). At some points in Figure 2, this is represented by arrows. The upper arrow at each point represents μ, the slope of the the path velocity evolution when the path acceleration, $\ddot{\lambda}$, is maximized. The lower arrow represents μ when the path acceleration, $\ddot{\lambda}$, is minimized. At each point of

Figure 2 the actual slope of the path velocity evolution should lie between these arrows. Note that on the border $T(\lambda)$ of the admissible region in Figure 2, the upper and lower arrows exactly overlap, as there is only one value of $\ddot{\lambda}$, and because of (25), only one value of μ is allowed.

In terms of Figure 2, the time-optimal control problem now comes down to maximizing the path velocity $\dot{\lambda}$ for each λ, $0 \leq \lambda \leq \lambda_{\max}$, where $\dot{\lambda}$ should stay inside the admissible region and satisfy the boundary conditions $\dot{\lambda}(0)$ and $\dot{\lambda}(t_{\max})$, and everywhere μ via (25) is bounded by (19). Shin and McKay (1985) developed an algorithm to maximize the path velocity $\dot{\lambda}$ in Figure 2 and thereby solved the time-optimal control problem if constraint (9) is disregarded. We want to solve the problem including constraint (9). Assuming from (18) that f_1 and g_1 are nonzero, we have

$$\dot{\lambda} \leq S(\lambda), \tag{26a}$$

where

$$S(\lambda) = \text{MIN}(s_{\max}/|f_1|, s_{\max}/|g_1|). \tag{26b}$$

The constraint (26) changes the admissible region in Figure 2 at points where $S(\lambda) < T(\lambda)$. We therefore redefine the border of the admissible region by $V(\lambda)$, which we again assume to be part of the admissible region. Obviously,

$$V(\lambda) = \text{MIN}(S(\lambda), T(\lambda)). \tag{27}$$

DEFINITION 1. An inadmissible point on the border $V(\lambda)$ of the admissible region is a point that does not allow continuation inside the admissible region of $\dot{\lambda}$ because of the bounds on μ. Accordingly, points on the border $V(\lambda)$ that are not inadmissible are called *admissible points on the border* $V(\lambda)$.

In Figure 2, $V(0.14)$ is an inadmissible point, whereas $V(0.5)$ is an admissible point on the border $V(\lambda)$.

Like Shin and McKay (1985), in the following we will limit the discussion to situations in which no "islands of inadmissibility" occur. However, our algorithm can be extended in exactly the same manner as theirs to include situations in which "islands of inadmissibility" do occur.

5. Solution to the Time-Optimal Control Problem

The solution of the problem is based on the following rule. When traveling forward in the λ domain, to maximize $\dot{\lambda}$ we should maximize μ, given by (25). From (19) and (25), observe that μ is maximized by maximizing the path acceleration,

$$\ddot{\lambda} = \beta_{\min}(\lambda, \dot{\lambda}). \tag{28}$$

Therefore the curve so constructed is called an *acceleration curve*.

DEFINITION 2. An *acceleration curve* is a curve started anywhere in the admissible region that travels forward in the λ domain while μ is maximized, except when on the border $V(\lambda)$. On the border $V(\lambda)$, μ is chosen to stay on the border [$V(\lambda)$, $0.25 < \lambda < 0.5$ in Figure 2]. When this is no longer possible, either this is an inadmissible point on $V(\lambda)$, where the acceleration curve ends [$V(0.14)$ in Figure 2], or the acceleration curve continues inside the admissible region where μ is maximized again [$V(0.5)$ in Figure 2]. If the acceleration curve does not meet an inadmissible point on $V(\lambda)$, it ends at the point where $\lambda = \lambda_{\max}$.

Traveling backward in the λ domain, to maximize $\dot{\lambda}$ we should minimize μ. From (19) and (26), observe that this is achieved by minimizing the path acceleration,

$$\ddot{\lambda} = \alpha_{\max}(\lambda, \dot{\lambda}). \tag{29*}$$

Accordingly, the curve so constructed is called a *deceleration curve*.

DEFINITION 3. A *deceleration curve* is a curve started anywhere inside the admissible region that travels *backward* in the λ domain while μ is minimized. It ends at either the first admissible point on $V(\lambda)$ it meets or at a point for which $\lambda = 0$.

Note that these definitions imply the following. Either two deceleration curves have no points in common or one includes the other. Furthermore, an acceleration curve and a deceleration curve have at most one point in common. Finally, note that traveling forward and backward in the λ domain, according to (13), is equivalent to traveling forward and backward in time.

DEFINITION 4. A point $(\lambda_1, \dot{\lambda}_1)$ in the $(\lambda, \dot{\lambda})$ plane *precedes* another point $(\lambda_2, \dot{\lambda}_2)$ if $\lambda_1 \leq \lambda_2$ (Note the equality.)

DEFINITION 5. An acceleration or deceleration curve *precedes* another acceleration or deceleration curve if the end point of the former precedes the end point of the latter (note that the end point of a deceleration curve precedes its starting point).

After application of the following algorithm, which consists of three steps, we obtain the solution to the time-optimal control problem.

Algorithm

1. From the initial values $\lambda = 0$ and $\dot{\lambda} = 0$, start an acceleration curve. If the acceleration curve ends on

* Equation 31 follows.

$V(\lambda)$, call this point $V(\lambda_a)$. By Definition 2, $V(\lambda_a)$ is an inadmissible point on $V(\lambda)$.

2. From the final values $\lambda = \lambda_{\max}$ and $\dot{\lambda} = 0$, start a deceleration curve. If the deceleration curve intersects the acceleration curve generated by step 1, this intersection is unique, and the procedure ends. Otherwise the deceleration curve ends at $V(\lambda)$. Call this point $V(\lambda_c)$. By Definition 3, $V(\lambda_c)$ is an admissible point on $V(\lambda)$.
3. Search the border $V(\lambda)$ from $V(\lambda_a)$ for the first admissible point $V(\lambda_b)$ on $V(\lambda)$, $\lambda_a < \lambda_b \leq \lambda_c$. From $V(\lambda_b)$, start a deceleration curve. Given the nature of the algorithm and Definitions 2 and 3, this curve intersects exactly one acceleration curve generated by the algorithm, so the computation may be stopped once an intersection is found. From $V(\lambda_b)$, start an acceleration curve. If this curve ends at an inadmissible point on $V(\lambda)$, this point becomes the new $V(\lambda_a)$, and this step is repeated. Otherwise this curve crosses the deceleration curve generated by step 2 and the procedure ends.

THEOREM 1. The set of acceleration and deceleration curves generated by the algorithm uniquely connects the initial and final conditions. This connection alternately consists of acceleration and deceleration curves and constitutes the solution to the time-optimal control problem.

Proof. The procedure ends if a curve intersects the deceleration curve generated by step 2. This has to be an acceleration curve that is generated by either step 1 or step 3. If the acceleration curve generated by step 1 intersects the deceleration curve generated by step 2, these curves uniquely connect the initial and final conditions. If the deceleration curve generated by step 2 is intersected by an acceleration curve generated by step 3, this acceleration curve started at $V(\lambda)$, where it was connected to a preceding deceleration curve that was connected to exactly one preceding acceleration curve generated by either step 1 or step 3. If this is the curve generated by step 1, again a unique connection is found; otherwise the acceleration curve started at $V(\lambda)$, where it was connected to a preceding deceleration curve, etc. Thus we have proved that the acceleration and deceleration curves generated by the algorithm alternate and uniquely connect the initial and final conditions.

Let Γ be the unique solution generated by the procedure. Note that Γ can be divided into sections consisting of one acceleration curve connected to one deceleration curve. All connection points should be considered part of the deceleration curve. Furthermore, note that the admissible region above each section is bounded by $\lambda = 0$, $\lambda = \lambda_{\max}$, $V(\lambda)$, or Γ itself. If the solution generated by our procedure is not optimal, a curve Γ' should exist with at least one point, say $(\lambda_0, \dot{\lambda}_0)$ above Γ. Now consider the section of Γ that contains λ_0. In this section, one point must exist where Γ' goes up from Γ and one point where Γ' comes down on Γ, since otherwise Γ' violates the boundary conditions or leaves the admissible region. Thus Γ' goes up from and comes down on the acceleration curve of the Γ section, Γ' goes up from and comes down on the deceleration curve of the Γ section, or Γ' goes up from the acceleration curve of the Γ section and comes down on the deceleration curve of the Γ section. However, Γ' going up from the acceleration curve of the Γ section violates the fact that μ on Γ is everywhere maximized; Γ' coming down on the deceleration curve of the Γ section violates the fact that μ on Γ is everywhere minimized, leaving no opportunities for Γ'. □

6. Computation of Time-Optimal Solutions

6.1. The Numerical Algorithm

We developed software to compute the time-optimal solution where f and g, which constitute the path (10), are obtained by cubic spline interpolation of a set of coordinate pairs

$$(x_i, y_i) \qquad i = 0, 1, 2, \ldots, M, \tag{31}$$

which constitute prescribed values of $x_p(t_i)$ and $y_p(t_i)$, where t_i is undetermined but with the property that $t_{i+1} > t_i$. Thus (31) constitutes a set of points in the X-Y plane that have to be passed in a given order. The function f is obtained by cubic spline interpolation of the sequence

$$x(\lambda_i), \qquad i = 0, 1, 2, \ldots, M, \tag{32a}$$

where

$$x(\lambda_i) = x_i, \qquad i = 0, 1, 2, \ldots, M \tag{32b}$$

$$\lambda_0 = 0, \tag{32c}$$

$$\lambda_{i+1} - \lambda_i = 1/M, \quad i = 0, 1, 2, \ldots, M-1. \tag{32d}$$

Accordingly, y is obtained by cubic spline interpolation of the sequence

$$y(\lambda_i), \qquad i = 0, 1, 2, \ldots, M, \tag{33a}$$

where

$$y(\lambda_i) = y_i, \qquad i = 0, 1, 2, \ldots, M. \tag{33b}$$

We chose Akima's cubic spline interpolation because it combats wiggles in the interpolant (De Boor 1978), and wiggles in the path are generally undesirable. We used the routine CSAKM from the IMSL library to perform this interpolation. Figure 3 shows the interpolation results

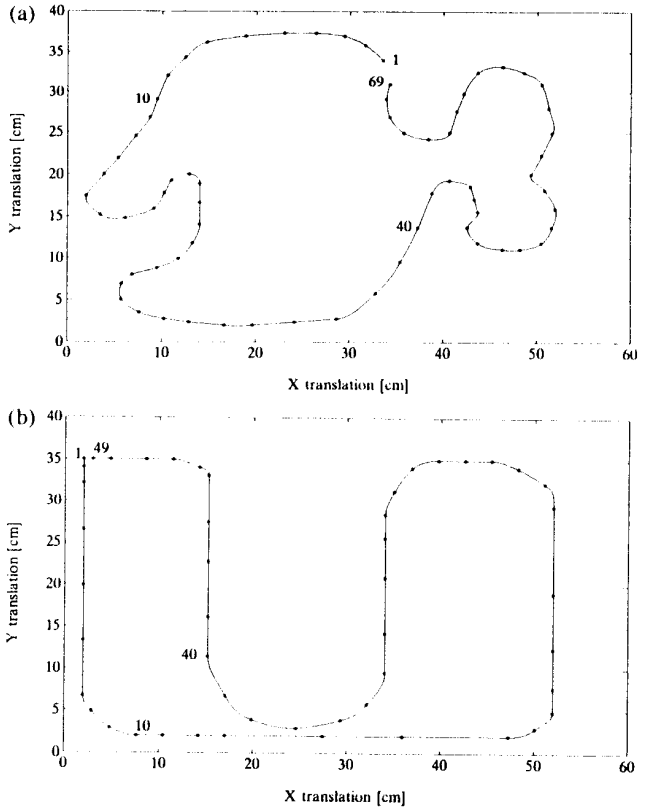

Fig. 3. Interpolated coordinate pairs. A, Path 1. B, Path 2.

(i.e., the paths) resulting from two sets of coordinate pairs numbered according to the order in which they have to be passed.

The computation of the time-optimal solution is largely based on the numerical integration of (28) and (29) forward and backward in time. $(\lambda_n, \dot{\lambda}_n, t_n)$ will refer to a point in the $(\lambda, \dot{\lambda})$ plane that is reached at time t_n. The numerical integration will be evaluated at equidistant times t_n; i.e.,

$$t_{n+1} - t_n = \Delta t \tag{34}$$

for acceleration curves and the search procedure, and

$$t_{n+1} - t_n = -\Delta t \tag{35}$$

for deceleration curves where Δt is a positive constant. For our application we chose

$$\Delta t = 1mS. \tag{36}$$

By definition, the numerical computation of an acceleration curve starts with $n = 0$, and so $(\lambda_0, \dot{\lambda}_0, t_0)$ is the starting point of the curve. The computational procedure consists of seven steps:

1. From $(\lambda_n, \dot{\lambda}_n, t_n)$, numerically integrate (28) forward in time and evaluate the result $(\lambda_{n+1}, \dot{\lambda}_{n+1}, t_{n+1})$.
2. If $\lambda_{n+1} > \lambda_{\max}$, stop the acceleration curve at $(\lambda_n, \dot{\lambda}_n, t_n)$.
3. If $\dot{\lambda}_{n+1} < V(\lambda_n)$, set $n = n + 1$, and go to 1.
4. If $V(\lambda_{n+1}) = T(\lambda_{n+1})$, stop the acceleration curve at $(\lambda_n, \dot{\lambda}_n, t_n)$.
5. Numerically integrate (29) from $(\lambda_n, \dot{\lambda}_n, t_n)$ forward in time and evaluate the result $(\lambda_{n+1}, \dot{\lambda}_{n+1}, t_{n+1})$.
6. If $\dot{\lambda}_{n+1} > S(\lambda_{n+1})$, stop the acceleration trajectory at $(\lambda_n, \dot{\lambda}_n, t_n)$.
7. Set $\dot{\lambda}_{n+1} = S(\lambda_{n+1})$, $n = n + 1$, and go to 1.

Steps 1 and 2 are the general steps taken. Step 3 involves the usual continuation if we are within the admissible region. Step 4 involves the situation in which an inadmissible point is reached on the border $V(\lambda) = T(\lambda)$. Step 5 is taken if we reached a point on the border $V(\lambda) = S(\lambda)$. Step 6 involves an inadmissible point on the border $V(\lambda) = S(\lambda)$, while step 7 involves the continuation on the border $V(\lambda) = S(\lambda)$.

By definition, the numerical computation of a deceleration curve starts with $n = 0$, so $(\lambda_0, \dot{\lambda}_0, t_0)$ is the starting point of the curve. The computational procedure consists of five steps:

1. Numerically integrate (29) from $(\lambda_n, \dot{\lambda}_n, t_n)$ backward in time and evaluate the result $(\lambda_{n+1}, \dot{\lambda}_{n+1}, t_{n+1})$.
2. If $\lambda_{n+1} < 0$, stop the deceleration curve at $(\lambda_n, \dot{\lambda}_n, t_n)$.
3. If $\dot{\lambda}_{n+1} > V(\lambda_n)$, stop the deceleration curve at $(\lambda_n, \dot{\lambda}_n, t_n)$.
4. If λ_{n+1} is inside the λ domain of an acceleration curve, search for the nearest computed value λ' in this domain and consider the point $(\lambda', \dot{\lambda}')$ of the acceleration curve. If $\dot{\lambda}_{n+1} > \dot{\lambda}'$, an intersection is found, and the deceleration curve ends at $(\lambda_n, \dot{\lambda}_n, t_n)$.
5. Set $n = n + 1$ and go to 1.

Steps 1 and 2 are the general steps taken. Step 3 involves the reaching of $V(\lambda)$. Step 4 involves the situation in which a connection to an acceleration curve is found.

The search for a next admissible point on $V(\lambda)$ is performed by the following procedure consisting of eight steps [by definition we start searching at $n = 0$ so at $(\lambda_0, \dot{\lambda}_0, t_0)$, where $\dot{\lambda}_0 = V(\lambda_0)$]:

1. Numerically integrate (28) from $(\lambda_n, \dot{\lambda}_n, t_n)$ forward in time and evaluate the result $(\lambda_{n+1}, \dot{\lambda}_{n+1}, t_{n+1})$.
2. If $\lambda_{n+1} > \lambda_{\max}$, stop the search.
3. If $\dot{\lambda}_{n+1} < V(\lambda_{n+1})$, $(\lambda_n, \dot{\lambda}_n)$ is the next admissible point on $V(\lambda)$.

4. If $V(\lambda_{n+1}) = T(\lambda_{n+1})$, go to step 8.
5. Numerically integrate (29) forward in time from $(\lambda_n, \dot{\lambda}_n, t_n)$ and evaluate the result $(\lambda_{n+1}, \dot{\lambda}_{n+1}, t_{n+1})$.
6. If $\dot{\lambda}_{n+1} < S(\lambda_{n+1})$, $(\lambda_n, \dot{\lambda}_n)$ is the next admissible point on $V(\lambda)$.
7. Set $\dot{\lambda}_{n+1} = S(\lambda_{n+1})$.
8. Set $n = n + 1$ and go to 1.

Steps 1 and 2 are the usual steps taken. Step 3 involves a continuation from the border inside the admissible region. Steps 5 through 8 are taken if the border $V(\lambda) = S(\lambda)$.

Because we obtain λ and $\dot{\lambda}$ through numerical integration of equation (28) forward and (29) backward in time, we immediately have available $\lambda(t)$ and $\dot{\lambda}(t)$. Via (12), (15a), and (15c), we therefore have available the state trajectory of the system (6). This state trajectory will be the basis for computing a digital controller.

The routine IVPRK from the IMSL library, which uses a Runge Kutta fifth- and sixth-order method, is used to perform the numerical integration. However, at some intervals, error conditions are obtained because equations (28) and (29) become stiff. Then a Euler integration routine was used in which very small steps were taken to ensure reasonable accuracy. We should mention that use of the Euler routine seriously decreases the computational efficiency. From (21)—which, via (19), determines $\alpha_{\max}$ and $\beta_{\min}$ in (28) and (29)—we observe that (28) and (29) become stiff if f_2 or g_2 becomes large or if both f_1 and g_1 become close to zero.

In Section 5 we assumed both f_1 and g_1 were nonzero everywhere. In practice, however, at some part of the path (10), one of the links stands still momentarily or for some time, and therefore f_1 or g_1 becomes zero at isolated points or intervals, respectively. If f_1 is zero, (26) is completely determined by $|g_1|$. From (16a) we obtain

$$\dot{\lambda}^2 = b_x u_x / f_2. \tag{37}$$

If f_1 is zero at an isolated point, f_2 is nonzero, (19b) determines the bounds on $\ddot{\lambda}$, (28) and (29) are well defined, and from (37) we have

$$T(\lambda) = |b_x u_{\max} / f_2|^{1/2}. \tag{38}$$

When g_1 is zero, (26) is completely determined by $|f_1|$, and from (16b) we have

$$\dot{\lambda}^2 = b_y u_y / g_2. \tag{39}$$

If g_1 is zero at an isolated point, g_2 is nonzero, (19a) determines the bounds on $\ddot{\lambda}$, (28) and (29) are again well defined, and from (39) we have

$$T(\lambda) = |b_y u_{\max} / g_2|^{1/2}. \tag{40}$$

Because we treat the case of isolated points, one may wonder if (38) and (40) result in a discontinuity of $T(\lambda)$, because this would have serious consequences for the numerical algorithm if $T(\lambda) < S(\lambda)$. However, this is not so.

LEMMA 1. Consider $f(\lambda_0)$ to be an isolated point where $f_1(\lambda_0) = 0$. Then

$$\lim_{\lambda \to \lambda_0} T(\lambda) = T(\lambda_0).$$

Proof. From (38) we have

$$T(\lambda_0) = |b_x u_{\max} / f_2(\lambda_0)|^{1/2}. \tag{41}$$

Because f is such that f_1 is continuous and $f(\lambda_0)$ is an isolated point where $f_1(\lambda_0) = 0$, then if $\lambda \to \lambda_0$, $f_1(\lambda) \to 0$. From (16a), given $f_1(\lambda) \to 0$, we obtain,

$$\begin{aligned} &(-b_x u_{\max} - f_2(\lambda)\dot{\lambda}^2 - c_x)/|f_1(\lambda)| \\ &\leq \ddot{\lambda} \\ &\leq (b_x u_{\max} - f_2(\lambda)\dot{\lambda}^2 + c_x)/|f_1(\lambda)|. \end{aligned}$$

Obviously, if $\dot{\lambda} > |b_x u_{\max} / f_2(\lambda)|^{1/2}$, the upper and lower bound both go to either ∞ or $-\infty$ as $f_1(\lambda) \to 0$. Together with (16b), this leaves no admissible values for $\ddot{\lambda}$. If $\dot{\lambda} < |b_x u_{\max} / f_2(\lambda)|^{1/2}$, the upper bound goes to $+\infty$ and the lower bound to $-\infty$ as $f_1(\lambda) \to 0$, with (16b) leaving admissible values of $\ddot{\lambda}$. □

For an isolated point $g(\lambda_0)$ where $g_1(\lambda_0) = 0$, we obtain analogous results. Although $T(\lambda)$ is continuous, it may rapidly change around λ_0, which may cause numerical problems.

Now consider the situation where $f_1 = 0$ over a closed interval $[\lambda_0, \lambda_1]$. Then for each $\lambda \in (\lambda_0, \lambda_1)$ $f_2(\lambda) = 0$, and (16a) imposes no bounds on $\dot{\lambda}$ or $\ddot{\lambda}$. Therefore, $T(\lambda)$ does not exist, which can be regarded as $T(\lambda) = \infty$. Therefore, $V(\lambda) = S(\lambda)$, which, by inspection of (26), is completely determined by $|g_1|$. If λ equals λ_0 or λ_1, this equals the situation of isolated points described previously. For $g_1 = 0$ over a closed interval, we obtain analogous results. From this we observe that the border $V(\lambda)$ at λ_0 and λ_1, the edges of the interval, may jump from $T(\lambda)$ to $S(\lambda)$, generally causing a discontinuity. To obtain a proper solution, $V(\lambda)$ should be defined as the smallest value of $\dot{\lambda}$ involved in the jump that constitutes an admissible point on $V(\lambda)$. It therefore is essential to continuously search the border for the next admissible point on it—i.e., to start the next integration step of the search procedure at the value of λ where the previous step ended.

Finally, if $f_1 = 0$ and $g_1 = 0$ simultaneously, this amounts to a situation where the path demands all links (i.e., the robot) to stand still, as can be seen from (15). We have already excluded this situation in Section 3.

Summarizing, we do allow f_1 and g_1 to be zero at intervals or isolated points, but not simultaneously. Furthermore, because we want to prescribe the initial and final state of the robot (6), from (15a) and (15c) we see that we do demand a path for which $f_1(0)$, $g_1(0)$, $f_1(\lambda_{\max})$, and $g_1(\lambda_{\max})$ are nonzero. In that case, $\dot{\lambda}(0)$, $\dot{\lambda}(t_{\max})$, and (12) uniquely determine the initial and final state of the robot (6). The main problem with the numerical computation of the algorithm is that the equations (28) and (29) may be stiff at some intervals. This causes the numerical integration to become computationally expensive and less accurate. This problem can be partially overcome by adjusting the parameterization (32) and (33) (i.e., by adjusting the values of λ_i).

6.2. *Numerical Examples*

The following parameter values applied to the industrial X-Y robot (Van Willigenburg 1991) if x_p, y_p (the link translations) are expressed in centimeters, $\dot{x}_p$, $\dot{y}_p$ (the link velocities) in cm/s, and $\ddot{x}_p$, $\ddot{y}_p$ (the link accelerations) in cm/s^2,

$$v_x = 0, \tag{42a}$$
$$v_y = 3.0, \tag{42b}$$
$$c_x = 70, \tag{42c}$$
$$c_y = 96, \tag{42d}$$
$$b_x = 70, \tag{42e}$$
$$b_y = 88, \tag{42f}$$
$$u_{\max} = 5.00, \tag{42g}$$
$$s_{\max} = 100.0. \tag{42h}$$

The bound (42g) should be chosen as high as possible, but leaving enough room for control corrections, which will be needed to deal with errors caused by uncertainty and imperfect modeling. The bound should therefore be experimentally determined and be less than the actual bound of the control.

From the numerical algorithm, the time-optimal state trajectory is available at equidistant times t_n the distance Δt given by (36). To check the accuracy of the computed time-optimal state trajectories, we computed the time-optimal control from these state trajectories and the robot dynamics (6) using the following approximations:

$$\ddot{x}_p(t_n) = (\dot{x}_p(t_{n+10}) - \dot{x}_p(t_n))/(10\Delta t), \tag{43a}$$
$$\ddot{y}_p(t_n) = (\dot{y}_p(t_{n+10}) - \dot{y}_p(t_n))/(10\Delta t), \tag{43b}$$

Using (6) and the approximation (43), we obtain for the control

$$u_x(t_n) = 1/b_x \left[\frac{(\dot{x}_p(t_{n+10}) - \dot{x}_p(t_n))}{(10\Delta t) + v_x \dot{x}_p(t_n) + c_x \operatorname{sign}(\dot{x}_p(t_n))} \right], \tag{44a}$$

$$u_y(t_n) = 1/b_y \left[\frac{(\dot{y}_p(t_{n+10}) - \dot{y}_p(t_n))}{(10\Delta t) + v_y \dot{y}_p(t_n) + c_y \operatorname{sign}(\dot{y}_p(t_n))} \right]. \tag{44b}$$

From Figure 4A we observe that, except for some peaks caused by the approximation (44) and by numerical integration errors at points where (28) and (29) are stiff, everywhere one control variable takes on an extreme value. This result is expected, since from Figure 4B we observe that the velocity bound (42h) is never reached, and therefore everywhere $\ddot{\lambda}$ takes on an extreme value.

Figure 5 shows the time-optimal solution for the path in Figure 3B. The values of the parameters in (42) again hold, except for (42g), which is replaced by

$$u_{\max} = 8.00. \tag{45}$$

From Figure 5, as expected, we observe that if one of the links moves with maximum velocity, none of the control variables takes on an extreme value. In the next section we present the computation and implementation of a digital optimal feedback controller based on the time-optimal state trajectories computed in this section, as well as experimental results after implementation.

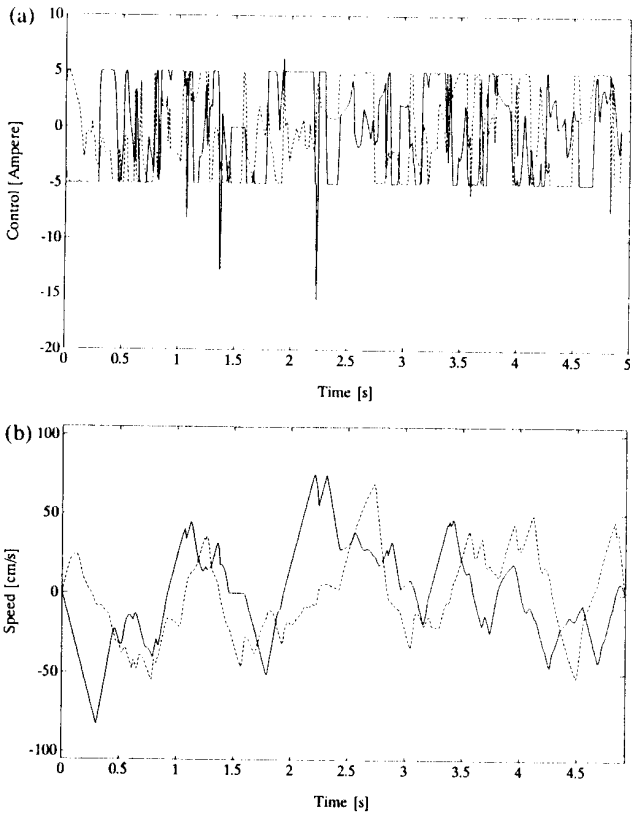

Fig. 4. A, Time-optimal control path 1. B, Time-optimal solution for path 1.

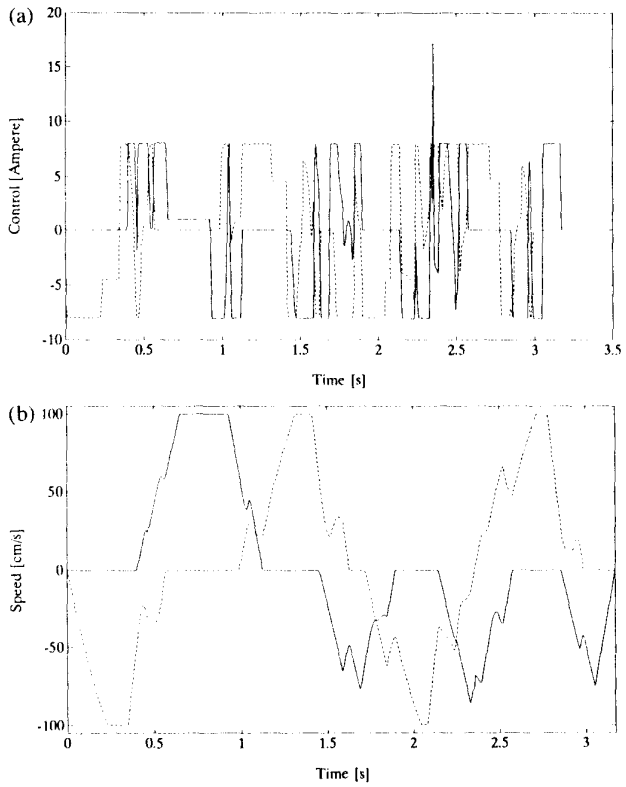

Fig. 5. A, Time-optimal control path 2. B, Time-optimal solution for path 2.

7. Computation and Implementation of a Digital Time-Optimal Feedback Controller

The X-Y robot is controlled by a digital computer. Therefore, measurements are only performed at certain so-called sampling instances, t_k, $k = 0, 1, 2, \ldots$, which are not necessarily equidistant. Based on the last measurement, a new control is computed and applied to the system. In between the sampling instances, the control remains unchanged, since a zero-order hold circuit connects the computer to the robot. Therefore, the control is constrained to be piecewise constant.

The robot dynamics (6) may be written as

$$\begin{bmatrix} \dot{x}_p \\ \dot{y}_p \\ \ddot{x}_p \\ \ddot{y}_p \end{bmatrix} = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & -v_x & 0 \\ 0 & 0 & -v_y & 0 \end{bmatrix} \begin{bmatrix} x_p \\ y_p \\ \dot{x}_p \\ \dot{y}_p \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ b_x & 0 \\ 0 & b_y \end{bmatrix} \begin{bmatrix} u_x \\ u_y \end{bmatrix} - \begin{bmatrix} 0 \\ 0 \\ \operatorname{sign}(\dot{x}_p)c_x \\ \operatorname{sign}(\dot{y}_p)c_y \end{bmatrix}. \quad (46)$$

If both links continue to move in the same direction from (46), we see that the nonlinear terms representing Coulomb friction can be compensated for by constant values of the control. Therefore, this compensation can be realized with a piecewise-constant control. Only if the direction of the motion of a link changes during a sample interval, the compensation cannot be realized by a piecewise-constant control. In this case we will not compensate for Coulomb friction. Therefore, in the following we will consider the linear system that remains if we disregard the last term in (46)—i.e.,

$$\dot{z} = A\,z + B\,u, \quad (47a)$$

where

$$z = \begin{bmatrix} x_p \\ y_p \\ \dot{x}_p \\ \dot{y}_p \end{bmatrix}, \quad (47b)$$

$$A = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & -v_x & 0 \\ 0 & 0 & -v_y & 0 \end{bmatrix}, \quad (47c)$$

$$B = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ b_x & 0 \\ 0 & b_y \end{bmatrix}, \quad (47d)$$

$$u = \begin{bmatrix} u_x \\ u_y \end{bmatrix}. \quad (47e)$$

The numerical solution of the digital LQ tracking problem for linear time-varying systems (Van Willigenburg 1991) enables us to solve the following optimal control problem. Given the linear system (47), its initial state $z(t_0)$, and an arbitrary reference state trajectory,

$$z_r(t) = \begin{bmatrix} x_{pr}(t) \\ y_{pr}(t) \\ \dot{x}_{pr}(t) \\ \dot{y}_{pr}(t) \end{bmatrix}, \qquad t \in [t_0, t_N], \quad (48)$$

find the optimal piecewise-constant control,

$$u(t) = u(t_k) \quad t \in [t_k, t_{k+1}), \quad k = 0, 1, 2, \ldots, N-1, \quad (49)$$

which minimizes

$$J(u) = (z(t_n) - z_r(t_n))^T H (z(t_n) - z_r(t_n)) + \int_{t_0}^{t_N} (z - z_r)^T Q(t)(z - z_r) + u^T R(t) u \, dt, \quad (50a)$$

where

$$Q(t) \geq 0 \qquad t \in [t_0, t_N], \quad (50b)$$

$$R(t) \geq 0 \qquad t \in [t_0, t_N], \quad (50c)$$

$$H \geq 0. \quad (50d)$$

The solution to this problem is given in feedback form,

$$u(t_k) = -\mathbf{L_k} x(t_k) + \mathbf{f_k}, \qquad k = 0, 1, 2, \ldots, N-1, \quad (51)$$

where $\mathbf{L_k}$ and $\mathbf{f_k}$ are precomputable feedback matrices and feedforward vectors, respectively.

We will assume a constant sampling interval—i.e.,

$$t_{k+1} - t_k = T, \qquad k = 0, 1, 2, \ldots, N-1, \tag{52}$$

where T is the sampling time, which equals 100 ms. Generally, a sampling interval of 100 ms is considered too large for proper robot control (Vukobratovic and Stokic 1982). However, the digital LQ tracker explicitly considers the intersample behavior and therefore results in a proper continuous-time behavior even for a relatively large sampling interval (Van Willigenburg 1991). The reference state trajectories $z_r(t)$ will be obtained after time scaling of the time-optimal state trajectories (12) computed in the previous section. The time scaling is such that $t_0 = 0$ while t_N is the nearest multiple of the sampling time T that is higher than $t_{\max}$. In the previous section we considered bounds on the control variables at this stage, and therefore we do not want to compromise on the choice of the control. Therefore, in (50) we choose

$$R(t) = 0. \tag{53}$$

Note that in general the continuous-time and the discrete-time LQ tracking problems (Lewis 1986), given (53), result in singular problems, while the digital LQ tracking problem, given (53), is generally nonsingular (Van Willigenburg 1991).

We are often only interested in deviations of the prescribed link positions. Therefore, a natural choice for $Q(t)$ is

$$Q(t) = \begin{bmatrix} q_1 & 0 & 0 & 0 \\ 0 & q_1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}, \tag{54}$$

where q_1 is a positive constant. At the final time, however, we generally want the robot to stand still, so a natural choice of H is

$$H = \begin{bmatrix} h_1 & 0 & 0 & 0 \\ 0 & h_1 & 0 & 0 \\ 0 & 0 & h_2 & 0 \\ 0 & 0 & 0 & h_2 \end{bmatrix}, \tag{55}$$

where, again, h_1 and h_2 are positive constants. Given (53), (54), and (55), the optimal digital control and corresponding state trajectory will depend on the ratios h_1/q_1 and h_1/h_2. Only the minimum cost will depend on the absolute values of q_1, h_1, and h_2. The following choice is made based on experiments and a compromise between the reaching of the final state and the tracking error that occurs:

$$q_1 = 1.0, \tag{56a}$$

$$h_1 = 0.3, \tag{56b}$$

$$h_2 = 0.1. \tag{56c}$$

We assume the initial state to match the reference state trajectory—i.e.,

$$z(t_0) = z_r(t_0). \tag{57}$$

This is a reasonable assumption, because before executing the motion, we control the system to the desired initial state $z_r(t_0)$. From equations (51), (46), and (48) and the rules to compensate for Coulomb friction, we finally obtain the optimal piecewise-constant control $u_0(t)$ for the system (46) given by,

$$u_0(t) = u_0(t_k) = -L_k x(t_k) + f_k + g_k, \quad t \in [t_k, t_{k+1}), \quad k = 0, 1, 2, \ldots, N-1, \tag{58a}$$

where

$$g_k = \begin{bmatrix} m_x \operatorname{sign}(\dot{x}_{pr}(t_k)) c_x / b_x \\ m_y \operatorname{sign}(\dot{y}_{pr}(t_k)) c_y / b_y \end{bmatrix}, \quad \mathrm{k} = 0, 1, 2, \ldots, \mathrm{N}-1, \tag{58b}$$

with

$$m_x = 1, \tag{58c}$$

if $\operatorname{sign}(\dot{x}_{pr}(t_k))$ equals $\operatorname{sign}(\dot{x}_{pr}(t_{k+1}))$; otherwise,

$$m_x = 0, \tag{58d}$$

and accordingly,

$$m_y = 1, \tag{58e}$$

if $\operatorname{sign}(\dot{y}_{pr}(t_k))$ equals $\operatorname{sign}(\dot{y}_{pr}(t_{k+1}))$; otherwise,

$$m_y = 0. \tag{58f}$$

From (58) we see that the control law is given in feedback form, while the feedback matrices $\mathbf{L_k}$ and the feedforward vectors $\mathbf{f_k}$ and $\mathbf{g_k}$ can all be computed a priori. The optimal state evolution corresponding to the control law (58), given the initial state $z(t_0) = z_r(t_0)$, we will denote by $z_0(t)$. Figure 6 shows the first two components of $z_0(t) - z_r(t)$, $t_0 \le t \le t_N$—i.e., the translation errors for the X and Y links caused by the piecewise-constant constraint on the control. Note that the error in practical situations is not caused just by the piecewise-constant nature of the control, but also by modeling and measurement errors and uncertainties. Figures 7 and 8 show deviations with respect to $z_0(t)$ that occurred during experiments with the implemented feedback control law (58) where the state $z(t)$ (i.e., the link translations and speeds) is measured. As can be seen, the errors resulting from the piecewise-constant constraint on the control and from modeling and measurement errors and uncertainties are of the same order. The actual bound on the control is given by

$$u_{\max} = 10.0. \tag{59}$$

Finally, Figure 9 shows the actual controls that resulted from the implemented control law (58). Obviously the bounds (42g) and (45) are violated. These bounds hold

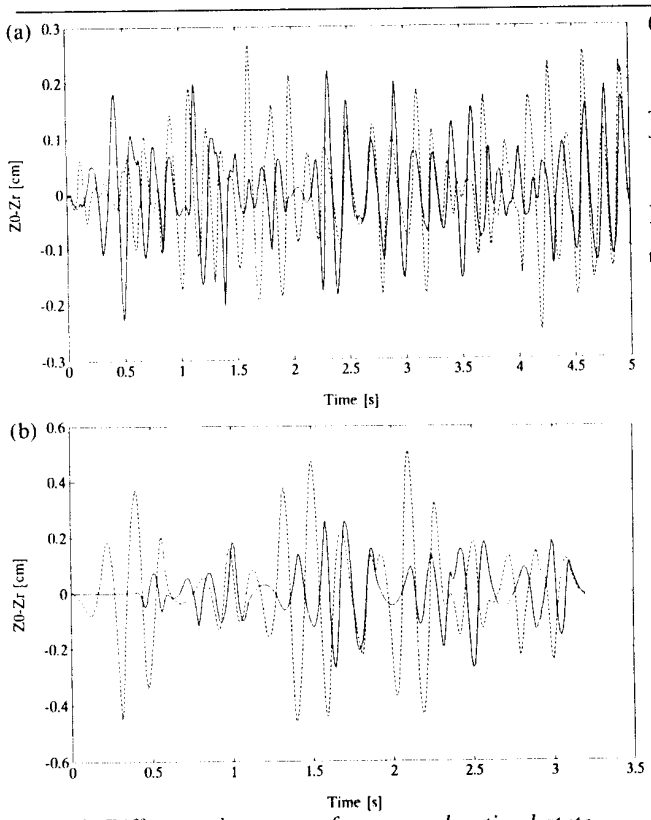

Fig. 6. Difference between reference and optimal state trajectory. A, Path 1. B, Path 2.

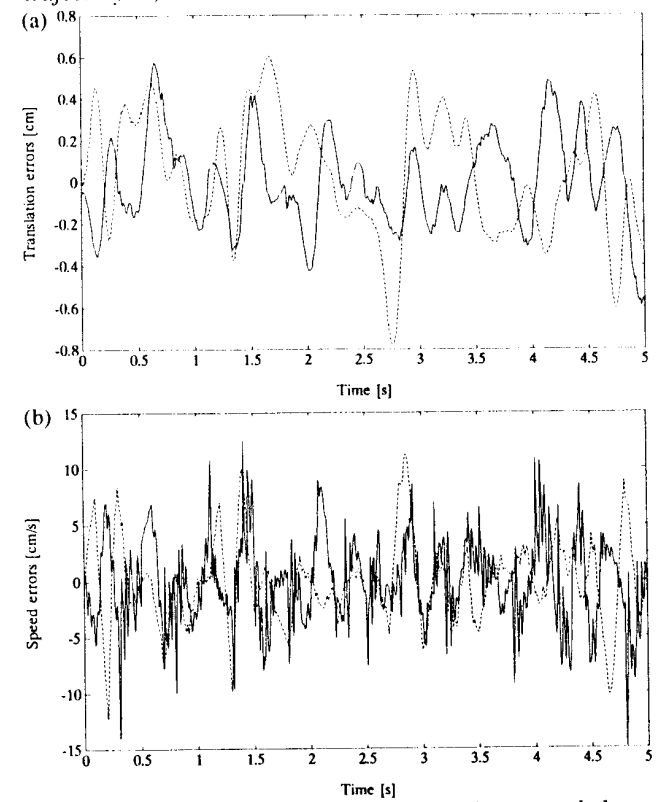

Fig. 7. A, Translation errors during experiment, path 1. B, Speed errors during experiment, path 1.

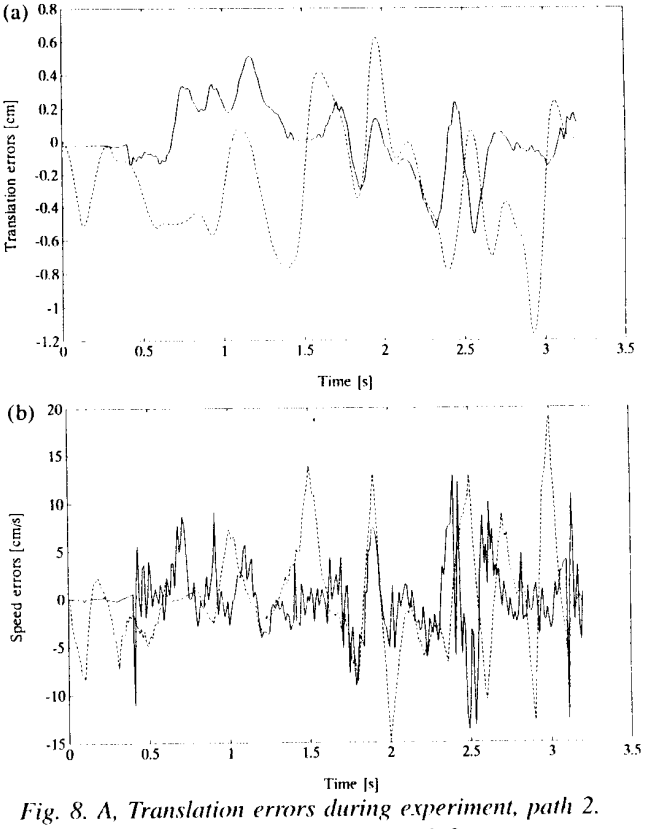

Fig. 8. A, Translation errors during experiment, path 2. B, Speed errors during experiment, path 2.

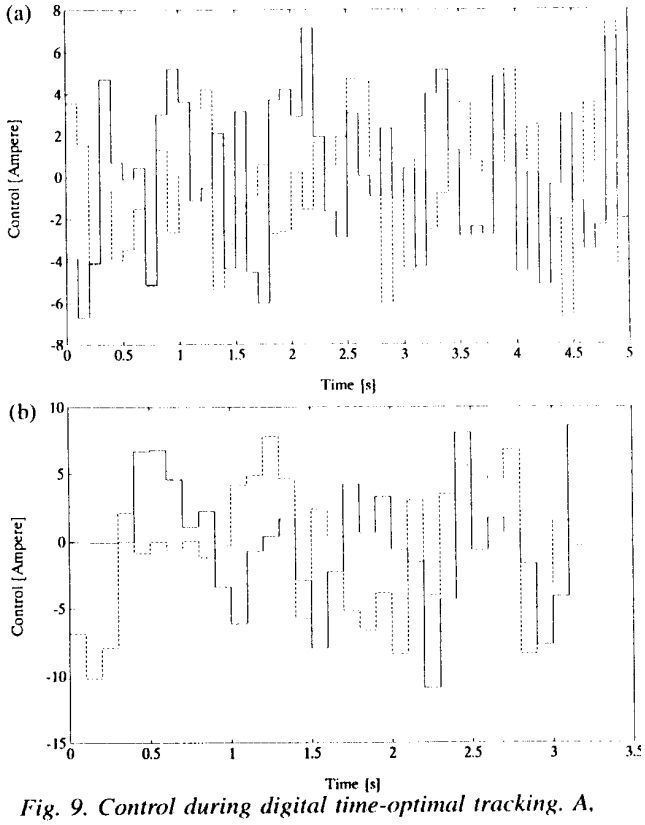

Fig. 9. Control during digital time-optimal tracking. A, Path 1. B, Path 2.

when the initial state and the model are exact and when we have a continuous-time control. The violation of these three requirements during application of the digital optimal control law (58) causes the control to violate the bounds (42g) and (45).

8. Conclusion

The computation of digital time-optimal feedback controllers for an industrial X-Y robot subjected to path constraints, actuation force constraints, and velocity limits for the individual links presented in this article consists of two parts. In the first part an optimal continuous-time, open-loop control was computed that achieves exact time-optimal tracking of the path if the initial state of the system and its dynamics are exact. In practice, because the X-Y robot is controlled by a digital computer, the control is piecewise constant and therefore does not allow exact tracking of the path. Furthermore, both the initial state and the model are not exact. Therefore, the second step constitutes the design of a digital LQ feedback controller that tracks the time-optimal state trajectory obtained from the first step as close as possible. Depending on the magnitude of modeling and initial state errors and the sampling interval, the control will deviate from the optimal continuous-time control and generally violate its bounds. Therefore, depending on the magnitude of modeling and initial state errors and the sampling interval, conservative bounds have to be chosen for the first part of the computation. One may wonder if it is possible to formulate the problem such that it can be solved at once. This is difficult, since exact tracking of the path is impossible while the desired system behavior is not specified as a function of time, making it very difficult to formulate an optimal control problem.

Our approach initially assumes an "ideal" situation for which the time-optimal tracking problem can be solved. It therefore allows us to compute the suboptimality caused by initial state and modeling errors and the magnitude of the sampling interval. This information helps us to decide on the choice of the sampling interval and improvement of the model.

References

Ailon, A., and Langholtz, G. 1985. On the existence of time-optimal control of mechanical manipulators. *J. Optimization Theory Applications* 46(1):1–20.

Bobrow, J. E., Dubowsky, S., and Gibson J. S. 1985. Time-optimal control of robotic manipulators along specified path. *Int. J. Robot. Res.* 4(3):3–17.

Craig, J. J. 1986. *Introduction to Robotics*. New York: Addison-Wesley.

De Boor, C. 1978. *A Practical Guide to Splines*. New York: Springer-Verlag.

Geering, H. P., Guzella, L., Hepner, S. A. R., and Onder, C. H. 1986. Time-optimal motions of robots in assembly tasks. *IEEE Trans. Auto. Control.* 31(6):512–518.

Lewis, F. L. 1986. *Optimal Control*. New York: John Wiley.

Shiller, Z., and Dubowsky, S. 1989. Robot path planning with obstacles: Actuator, gripper, and payload constraints. *Int. J. Robot. Res.* 8(6):3–18.

Shin, K. G., and McKay, N. 1985. Minimum-time control of robotic manipulators with geometric path constraints. *IEEE Trans. Auto. Control* AC-30(6):531–541.

Sontag, E. D., and Sussman, H. J. 1986. Time-optimal control of manipulators. *IEEE Conf. Robot. Automation*, San Francisco, pp. 1692–1697.

Vukobratovic, M., and Stokic, D. 1982.*Control of Manipulation Robots: Theory and Application*. Berlin: Springer-Verlag.

Van Willigenburg, L. G. 1991. *Digital optimal control of rigid manipulators*. Ph.D. thesis, Delft University Press, The Netherlands.