Minimum-Time Control of Robotic Manipulators with Geometric Path Constraints

KANG. G. SHIN, SENIOR MEMBER, IEEE, AND NEIL D. McKAY

Abstract—**Conventionally, robot control algorithms are divided into two stages, namely, *path* or *trajectory planning* and *path tracking* (or *path control*). This division has been adopted mainly as a means of alleviating difficulties in dealing with complex, coupled manipulator dynamics. Trajectory planning usually determines the timing of manipulator position and velocity *without* considering its dynamics. Consequently, the simplicity obtained from the division comes at the expense of efficiency in utilizing robot's capabilities.**

To remove at least partially this inefficiency, this paper considers a solution to the problem of moving a manipulator in minimum time along a specified geometric path subject to input torque/force constraints. We first describe the manipulator dynamics using parametric functions which represent geometric path constraints to be honored for collision avoidance as well as task requirements. Second, constraints on input torques/forces are converted to those on the parameters. Third, the minimum-time solution is deduced in an algorithm form using phase-plane techniques. Finally, numerical examples are presented to demonstrate utility of the trajectory planning method developed.

I. Introduction

INDUSTRIAL robots have emerged as a primary means of contemporary automation due to their potential for productivity increase and product quality improvement. Obviously, a robot should be controlled so as to produce as many units as possible per dollar invested. This in turn naturally leads to the need for minimum-time control of robots.

There are a variety of algorithms available for robot control. These algorithms usually assume that the control structure of the robot has been divided into two levels. The lower level is called *control* or *path tracking,* and the upper level is called *path* or *trajectory planning*. The path tracker attempts to make the robot's actual position and velocity match some desired values of position and velocity; the desired values are provided to the controller by the trajectory planner. The trajectory planner receives as input some sort of geometric path descriptor from which it calculates a time history of the desired positions and velocities. The path tracker then tries to minimize the deviation of the actual position and velocity from the desired values.

The control scheme is divided in this way because the process of robot control, if considered in its entirety, is very complicated, since the dynamics of all but the simplest robots are highly nonlinear and coupled. Dividing the controller into the two parts makes the whole process simpler. The path tracker is frequently a linear controller (e.g., a PID controller). While the nonlinearities of manipulator dynamics frequently are not taken into account at this level, such trackers can generally keep the manipulator fairly close to the desired trajectory.

Unfortunately, the simplicity obtained from the division into trajectory planning and path tracking comes at the expense of efficiency. The source of the inefficiency is the trajectory planner. In order to use the robot at maximum efficiency, the trajectory planner must be aware of the robot's dynamic properties, and the more accurate the dynamic model is, the better the robot's capabilities can be used. However, most of the trajectory planning algorithms presented to date assume very little about the robot's dynamics. The usual assumption is that there are constant or piecewise constant bounds on the robots velocity and acceleration [9], [10]. In fact, these bounds vary with position, payload mass, and even with payload shape. Thus, in order to make the constant-upper-bound scheme work, the upper bounds must be chosen to be global greatest lower bounds of the velocity and acceleration values; in other words, the worst case limits have to be used. Since the moments of inertia seen at the joints of the robot, and hence the acceleration limits, may vary by a factor of three or more, such bounds can result in considerable inefficiency or underutilization of the robot.

To alleviate the inefficiency, this paper presents a solution to the minimum-time manipulator control problem subject to constraints on its geometric path and input torques/forces. The solution will be in the form of a trajectory planning algorithm, and will take into account the details of the dynamics of the manipulator. The output of the trajectory planner will be the true minimum-time solution, and so will be useful as a standard against which the performance of other trajectory planning algorithms may be measured. Note that the problem and its solution considered in this paper are different from the near minimum-time control methods in [4], [5].

Bobrow *et al.* [1], [2] have independently come to conclusions similar to our own. Although their formulation of the problem is similar to ours, their solution algorithm and motivations are different from ours in several respects. In their work, no specific form is assumed for the actuator torque bounds except that they be functions only of the robot's current position and velocity. We have assumed that the velocity dependence is at most quadratic, which allows a more concrete treatment of some aspects of the problem while still allowing treatment of most of the actuators found in practice. The assumption of quadratic velocity bounds combined with the assumption that the parametric equations of the curve to be followed are piecewise analytic (again, an assumption which presents no practical difficulties) permits construction of a proof that our algorithm terminates in a finite number of steps. By contrast, Bobrow's work relies on the hypothesis that there are a finite number of switching points. Finally, both [1], [2], this paper make use of phase plane techniques, and both use the idea of a set of admissible velocities for a given position. This naturally leads to the idea of an "admissible region" in the phase plane. Bobrow's solution implicitly assumes that this region is simply connected. However, we show in this paper that 1) the admissible region may *not* be simply connected, possibly making some steps of Bobrow's algorithm impossible, 2) our algorithm terminates within a finite number of steps even for nonsimply connected admissible regions, and 3) the resulting algorithm is optimal for the general case, i.e., for nonsimply connected admissible regions.

The remainder of this paper is divided into five sections.

Manuscript received August 4, 1983; revised November 26, 1983 and April 20, 1984. Paper recommended by Past Associate Editor, J. Y. S. Luh. This work was supported in part by the National Science Foundation under Grant ECS 8409938, the U.S. Air Force Office of Scientific Research under Contract F49620-82-C-0089 and the Robot Systems Division, Center for Research and Integrated Manufacturing (CRIM), The University of Michigan, Ann Arbor, MI.

The authors are with the Department of Electrical Engineering and Computer Science, The University of Michigan, Ann Arbor, MI 48109.

0018-9286/85/0600-0531$01.00

Section II describes a method for making the manipulator dynamic equations more tractable and a method for handling input torque constraints. Section III contains a detailed formulation of the minimum-time control problem. Also, the form of the optimal solution is deduced using phase-plane techniques. Section IV presents an algorithm for generating minimum-time trajectories, and proofs of the convergence of the algorithm and optimality of the generated trajectories. In Section V, we discuss some simple examples to demonstrate the utility of our trajectory planning algorithm. The final section discusses the significance of the results.

II. Parameterized Robot Dynamics with Input Torque Constraints

Before delving headlong into the problem of minimum-time control, a dynamic model of the manipulator is required. There are a number of ways of obtaining the dynamic equations of a robot arm, i.e., the equations which relate joint forces and torques to positions, velocities, and accelerations. The Lagrange formulation of mechanism dynamics yields a set of differential equations which are easy to manipulate for robot control problems, and so will be used here [3], [11]. The dynamic equations take the form

$$u_i = J_{ij}(q)\ddot{q}^j + R_{ij}\dot{q}^j + C_{ijk}(q)\dot{q}^j\dot{q}^k + G_i(q) \tag{1}$$

where u_i is the ith generalized force, q^i is the ith generalized coordinate, J_{ij} the inertia matrix, G_i the gravitational force on the ith joint, C_{ijk} the Coriolis force array, and R_{ij} is the viscous friction matrix. The Einstein summation convention has been used, and all indexes run from one to n inclusive for an n-degree-of-freedom robot.

The motion of the robot arm will not, of course, be completely unconstrained. In fact, it will later be assumed that the manipulator must be constrained to a fixed path in joint space, and that the path is given as a *parameterized curve*. The curve is assumed to be given by a set of n functions of a single parameter λ, so that we are given

$$q^i = f^i(\lambda), \qquad 0 \le \lambda \le \lambda_{\max} \tag{2}$$

where λ is a parameter for describing the desired path, and it is assumed that the coordinates q^i vary continuously with λ and that the path never retraces itself as λ goes from 0 to $\lambda_{\max}$.

It should be noted that in practice the spatial paths are given in Cartesian coordinates. While it is in general difficult to convert a curve in Cartesian coordinates to that in joint coordinates, it is relatively easy to perform the conversion for individual points. One can then pick a sufficiently large number of points on the Cartesian path, convert to joint coordinates, and use some sort of interpolation technique (e.g., cubic splines) to obtain a similar path in joint space (see [8] for an example).

Returning to the problem at hand, we may use the parameterization of the q^i and differentiate with respect to time, giving

$$\dot{q}^i = \frac{df^i}{d\lambda}\frac{d\lambda}{dt} = \frac{df^i}{d\lambda}\dot{\lambda} = \frac{df^i}{d\lambda}\mu \tag{3}$$

where $\mu \equiv \dot{\lambda}$. The equations of motion along the curve (i.e., the geometric path) then become

$$\dot{\lambda} = \mu \tag{4a}$$

$$u_i = J_{ij}(\lambda)\frac{df^j}{d\lambda}\dot{\mu} + J_{ij}(\lambda)\frac{d^2f^j}{d\lambda^2}\mu^2 + G_i(\lambda) + R_{ij}\frac{df^j}{d\lambda}\mu + C_{ijk}(\lambda)\frac{df^j}{d\lambda}\frac{df^k}{d\lambda}\mu^2. \tag{4b}$$

Note that if λ is used to represent arc length along the path, then μ and $\dot{\mu}$ are the velocity and the acceleration along the path, respectively.

With this parameterization, there are two state variables, i.e., λ and μ, but $(n + 1)$ equations. One way to look at the system is to choose the equation $\dot{\lambda} = \mu$ and one of the remaining equations as state equations, regarding the other equations as constraints on the inputs and on $\dot{\mu}$. However, the problem has a more appealing symmetry if a single differential equation is obtained from the n equations given by multiplying the ith equation by $df^i/d\lambda$ and sum over i, giving

$$u_i\frac{df^i}{d\lambda} = J_{ij}(\lambda)\frac{df^i}{d\lambda}\frac{df^j}{d\lambda}\dot{\mu} + J_{ij}(\lambda)\frac{df^i}{d\lambda}\frac{d^2f^j}{d\lambda^2}\mu^2 + G_i(\lambda)\frac{df^i}{d\lambda} + R_{ij}\frac{df^i}{d\lambda}\frac{df^j}{d\lambda}\mu + C_{ijk}(\lambda)\frac{df^i}{d\lambda}\frac{df^j}{d\lambda}\frac{df^k}{d\lambda}\mu^2. \tag{5}$$

This formulation has a distinct advantage. Note that the coefficient of $\dot{\mu}$ is quadratic in the vector of derivatives of the constraint functions. Since a smooth curve[1] can always be parameterized in such a way that the first derivatives never all disappear simultaneously, and since the inertia matrix is positive definite, the whole equation can be divided by the nonzero positive coefficient of $\dot{\mu}$, providing a solution for $\dot{\mu}$ in terms of λ and μ. Now there are only two state equations, and the original n equations can be regarded as constraints on the inputs and on $\dot{\mu}$ (more on this will be discussed later).

With this formulation, the state equations become

$$\dot{\lambda} = \mu \tag{6a}$$

$$\dot{\mu} = \frac{1}{J_{ij}(\lambda)\dfrac{df^i}{d\lambda}\dfrac{df^j}{d\lambda}}\left[u_i\frac{df^i}{d\lambda} - J_{ij}(\lambda)\frac{df^i}{d\lambda}\frac{d^2f^j}{d\lambda^2}\mu^2 - G_i(\lambda)\frac{df^i}{d\lambda} - R_{ij}\frac{df^i}{d\lambda}\frac{df^j}{d\lambda}\mu - C_{ijk}(\lambda)\frac{df^i}{d\lambda}\frac{df^j}{d\lambda}\frac{df^k}{d\lambda}\mu^2\right]. \tag{6b}$$

Consider now the constraints on the inputs, namely, $u^i_{\min} \le u_i \le u^i_{\max}$ and (4b). The dynamic equations (4b) can be viewed as having the following form: $u_i = g_i(\lambda)\dot{\mu} + h_i(\lambda, \mu)$. For a given state, i.e., given λ and μ, this is just a set of parametric equations for a line, where the parameter is $\dot{\mu}$. The admissible controls, then, are those which are on this line in the input space and also are inside the rectangular prism formed by the input magnitude constraints. Thus, the rectangular prism puts bounds on $\dot{\mu}$. The reason for converting from bounds on the input torques/forces to bounds on the *pseudoacceleration* $\dot{\mu}$ is that all the positions, velocities, and accelerations of the various joints are related to one another through the parameterization of the path. Given the current state (λ, μ), the quantity $\dot{\mu}$, if known, determines the input torques/forces for *all* of the joints of the robot, so that manipulation of this one scalar quantity can replace the manipulation of n scalars (the input torques) and a set of constraints (the path parameterization equations).

For evaluating the bounds on $\dot{\mu}$ explicitly, (4b) can be plugged into the inequalities $u^i_{\min} \le u_i \le u^i_{\max}$ so that

$$u^i_{\min} \le J_{ij}\frac{df^j}{d\lambda}\dot{\mu} + \left(J_{ij}\frac{d^2f^j}{d\lambda^2} + C_{ijk}\frac{df^j}{d\lambda}\frac{df^k}{d\lambda}\right)\mu^2 + R_{ij}\frac{df^j}{d\lambda}\mu + G_i \le u^i_{\max}. \tag{7a}$$

[1] We assumed to have a smooth curve for describing the given path between the starting and destination points. If it is not, still we can divide the path into smooth subpaths. Then, the above assumption becomes valid.

Introducing some shorthand notation, let

$$M_i \equiv J_{ij}\frac{df^j}{d\lambda},\quad Q_i \equiv J_{ij}\frac{d^2f^j}{d\lambda^2}+C_{ijk}\frac{df^j}{d\lambda}\frac{df^k}{d\lambda},\quad R_i \equiv R_{ij}\frac{df^j}{d\lambda},\quad S_i \equiv G_i.$$

We then have

$$u^i_{\min} \le M_i\dot{\mu} + Q_i\mu^2 + R_i\mu + S_i \le u^i_{\max}. \tag{7b}$$

Note that the quantities listed above are functions of λ. For the sake of brevity, the functional dependence is not indicated in what follows.

Manipulation of these inequalities gives (assuming that $M_i \neq 0$)

$$\frac{u^i_{\min} - Q_i\mu^2 - R_i\mu - S_i}{|M_i|} \le \operatorname{sgn}(M_i)\dot{\mu} \le \frac{u^i_{\max} - Q_i\mu^2 - R_i\mu - S_i}{|M_i|}$$

$$\text{or } LB_i \le \dot{\mu} \le UB_i \tag{8}$$

where

$$LB_i \equiv \frac{u^i_{\min}(M_i>0) + u^i_{\max}(M_i<0) - (Q_i\mu^2 + R_i\mu + S_i)}{M_i}$$

and

$$UB_i \equiv \frac{u^i_{\max}(M_i>0) + u^i_{\min}(M_i<0) - (Q_i\mu^2 + R_i\mu + S_i)}{M_i}.$$

The expression $(M_i > 0)$ evaluates to one if $M_i > 0$, zero otherwise. Since these constraints must hold for all n joints, $\dot{\mu}$ must satisfy $\max_i LB_i \le \dot{\mu} \le \min_i UB_i$, or $GLB(\lambda, \mu) \le \dot{\mu} \le LUB(\lambda, \mu)$.

Note that it has *not* been assumed here that $u^i_{\min}$ and $u^i_{\max}$ are constants; they may indeed be arbitrary functions of λ and μ. Later these quantities will be assumed to have specific, relatively simple forms, but these forms should be adequate to describe most of the actuators used in practice.

The difference between the trajectory planning algorithm to be presented and those which are conventionally used can be seen in terms of the equation above. Assume that the parameter λ is arc length in Cartesian space. Then μ is the speed and $\dot{\mu}$ the acceleration along the geometric path. Since most conventional trajectory planners put *constant* bounds on the acceleration over some particular (frequently the entire) interval, one would have $GLB(\lambda, \mu) \le \dot{\mu}_{\min} \le \dot{\mu} \le \dot{\mu}_{\max} \le LUB(\lambda, \mu)$, where $\dot{\mu}_{\min}$ and $\dot{\mu}_{\max}$ are constants. The conventional techniques, then, restrict the acceleration more than is really necessary. Likewise, constant bounds on the velocity will also be more restrictive than necessary.

III. Formulation of Optimal Control Problem

With the manipulator dynamic equations and joint torque/force constraints in suitable form, we can address the actual control problem. Problems which require the minimization of cost functions subject to differential equation constraints can be expressed very naturally in the language of optimal control theory. The usual method of solving such a problem is to employ Pontryagin's maximum principle [6]. The maximum principle yields a two-point boundary value problem which is, except in some simple cases, impossible to solve in closed form, and may be difficult to solve numerically as well. We will therefore not use the maximum principle, but will use some simpler reasoning, taking advantage of the specific form of the cost function and of the controlled system.

In the case considered here, minimum cost is equated with minimum time, thus maximizing the operating speed of the robot. The cost function can then be expressed as $T = \int_0^{t_f} 1 \cdot dt$ where the final time t_f is left free. It is assumed here that the desired geometric path of the manipulator has been preplanned,[2] and is provided to the minimum-time controller in *parametric form,* as described earlier [i.e., (3)]. Further, assume that the q^i are parameterized in such a way that the initial point corresponds to $\lambda = 0$, the final point corresponds to $\lambda = \lambda_{\max}$, and that the $df^i/d\lambda$ never all become zero simultaneously. This guarantees that the state equations (6a) and (6b) exist, and also guarantees that as λ increases from 0 to $\lambda_{\max}$ the path never retraces itself.

Given this form for the dynamic equations, we have the minimum time path planning (MTPP) problem as follows.

Problem MTPP: Find $x^* = (\lambda^*, \mu^*)$ and u^*_i by minimizing T subject to (6a), (6b), $u^i_{\min} \le u_i \le u^i_{\max}$, $0 \le \lambda \le \lambda_{\max}$, and the boundary conditions $\mu(0) = \mu_0$, $\mu(t_f) = \mu_f$, $\lambda(0) = 0$, and $\lambda(t_f) = \lambda_{\max}$.

Before doing any further manipulations on the state equations, define the functions.

$$M(\lambda) \equiv J_{ij}(\lambda)\frac{df^i}{d\lambda}\frac{df^j}{d\lambda},$$

$$Q(\lambda) \equiv J_{ij}(\lambda)\frac{df^i}{d\lambda}\frac{d^2f^j}{d\lambda^2} + C_{ijk}(\lambda)\frac{df^i}{d\lambda}\frac{df^j}{d\lambda}\frac{df^k}{d\lambda},$$

$$R(\lambda) \equiv R_{ij}\frac{df^i}{d\lambda}\frac{df^j}{d\lambda},\quad S(\lambda) \equiv G_i(\lambda)\frac{df^i}{d\lambda},\text{ and }\quad U(\lambda) \equiv u_i\frac{df^i}{d\lambda}.$$

Again, for convenience the dependence of the above coefficients on λ will be omitted in the sequel. Now rewrite the state equations in the following form:

$$\dot{\lambda} = \mu \tag{9a}$$

$$\dot{\mu} = \frac{1}{M}[U - Q\mu^2 - R\mu - S]. \tag{9b}$$

The M term is a quadratic form reminiscent of the expression for the manipulator's kinetic energy. In fact, if the parametric expressions for the $\dot{q}_i$ are plugged into the formula for kinetic energy, one obtains the expression $2K = M\mu^2$. The Q term represents the components of the Coriolis and centrifugal forces which act along the path plus the fictitious forces generated by the restriction that the robot stay on the parameterized path. The R term represents frictional components, and S gives the gravitational force along the path. U is the projection of the input vector onto the velocity vector.

At this point, it is instructive to look at the system's behavior in the phase plane. The equations of the phase-plane trajectories can be obtained by dividing (9b) by (9a). This gives

$$\frac{d\mu}{d\lambda} = \frac{\dfrac{d\mu}{dt}}{\dfrac{d\lambda}{dt}} = \frac{\dot{\mu}}{\mu} = \frac{1}{\mu M}[U - Q\mu^2 - R\mu - S]. \tag{10}$$

It is interesting to note that the total time T it takes to go from initial to final states is

$$T = \int_0^{t_f} dt = \int_0^{\lambda_{\max}} \frac{dt}{d\lambda}\,d\lambda = \int_0^{\lambda_{\max}} \frac{1}{\mu}\,d\lambda. \tag{11}$$

The idea, then, is to minimize this integral subject to the given

[2] This is done at the stage of task planning to avoid collision as well as to meet task requirements.

constraints. We therefore want to make the pseudovelocity μ as large as possible, a result which would be expected intuitively.

The constraints on $\dot{\mu}$ have two effects. One effect is to place limits on the slope of the phase trajectory. The other is to place limits on the value of μ. To obtain the limits on $d\mu/d\lambda$, one simply divides the limits on $\dot{\mu}$ by μ, since $d\mu/d\lambda = \dot{\mu}/\mu$.

To get the constraints on μ, it is necessary to consider the bounds on $\dot{\mu}$. If, for particular values of λ and μ, we have $LUB(\lambda, \mu) < GLB(\lambda, \mu)$ then there are no permissible values of $\dot{\mu}$. Therefore, for each value of λ we can assign a set of values of μ as determined by the inequality $LUB(\lambda, \mu) - GLB(\lambda, \mu) \geq 0$. This inequality holds if and only if $UB_i(\lambda, \mu) - LB_j(\lambda, \mu) \geq 0$ for all i and j. The intersection of the regions determined by these inequalities produces a region of the phase outside of which the phase trajectory must not stray. This region will hereafter be referred to as the *admissible region* of the phase plane. Using the equations for the lower and upper bounds for all i and j,

$$\frac{u^i_{\max}(M_i>0)+u^i_{\min}(M_i<0)-(Q_i\mu^2+R_i\mu+S_i)}{M_i}$$

$$-\frac{u^j_{\min}(M_j>0)+u^j_{\max}(M_j<0)-(Q_j\mu^2+R_j\mu+S_j)}{M_j}\geq 0.$$

Rearranging this inequality,

$$\left[\frac{Q_i}{M_i}-\frac{Q_j}{M_j}\right]\mu^2+\left[\frac{R_i}{M_i}-\frac{R_j}{M_j}\right]\mu\left[\frac{S_i}{M_i}-\frac{S_j}{M_j}\right]$$

$$+\left[\frac{u^i_{\max}(M_i<0)-u^i_{\min}(M_i>0)}{|M_i|}\right.$$

$$\left.-\frac{u^i_{\max}(M_i<0)-u^i_{\min}(M_i>0)}{|M_j|}\right]\geq 0. \tag{12a}$$

It will prove convenient to "symmetrize" the input torque bounds in the discussion which follows. Each joint has a mean torque u^i_M and a maximum deviation Δ^i given by

$$u^i_M\equiv\frac{u^i_{\max}+u^i_{\min}}{2},\quad \Delta^i\equiv\frac{u^i_{\max}-u^i_{\min}}{2}.$$

The inequality (12a) can then be rewritten as

$$\left[\frac{Q_i}{M_i}-\frac{Q_j}{M_j}\right]\mu^2+\left[\frac{R_i}{M_i}-\frac{R_j}{M_j}\right]\mu+\left[\frac{S_i}{M_i}-\frac{S_j}{M_j}\right]$$

$$-\left[\frac{u^i_M}{M_i}-\frac{u^i_M}{M_j}\right]+\left[\frac{\Delta^i}{|M_i|}+\frac{\Delta^j}{|M_j|}\right]\geq 0. \tag{12b}$$

At this point, a specific form for the torque bounds will be assumed. If the maximum and minimum torques for each joint are functions only of the states q^i and $\dot{q}^i$ (i.e., the actuator torques are all independent of one another) and are at most *quadratic* in the velocities $\dot{q}^i$, then the inequality yields a simple quadratic in μ. This allows one to solve for the velocity bounds using the quadratic formula. A particulary simple and useful special case is that encountered when the actuator is a fixed-field DC motor with a bounded voltage input. In this case, the torque constraints take the form $u^i_{\max} = V^i_{\max} + k^i_m\dot{q}^i$ and $\dot{u}^i_{\min} = V^i_{\min} + k^i_m\dot{q}^i$ where $V^i_{\min}$ and $V^i_{\max}$ are proportional to the voltage limits and k^i_m is a constant which depends upon the motor winding resistance, voltage source resistance, and back E.M.F. generated by the motor. Let $V_{\text{ave}} = (V^i_{\max} + V^i_{\min})/2$ and $\Delta^i = V^i_{\max} - V^i_{\min}$. Then we get

$$u^i_M = V_{\text{ave}} + k^i_m\dot{q}^i = V_{\text{ave}} + k^i_m\frac{df^i}{d\lambda}\mu.$$

From here on, the case outlined above will be used for the sake of simplicity. The only changes required for the more general case of quadratic velocity dependence of the torque bounds is a redefinition of the coefficients in some the equations which follow.

Introducing yet more shorthand notation, let

$$A_{ij}\equiv\left[\frac{Q_i}{M_i}-\frac{Q_j}{M_j}\right]\quad B_{ij}\equiv\left[\frac{R_i}{M_i}-\frac{R_j}{M_j}\right]$$

$$-\left[\frac{V_{\text{ave}}+k^i_m\frac{df^i}{d\lambda}\mu}{M_i}-\frac{V_{\text{ave}}+k^j_m\frac{df^j}{d\lambda}\mu}{M_j}\right]$$

$$C_{ij}\equiv\left[\frac{\Delta^i}{|M_i|}+\frac{\Delta^j}{|M_j|}\right]\quad D_{ij}\equiv\left[\frac{S_j}{M_j}-\frac{S_i}{M_i}\right].$$

Noting that (at least in this case), $A_{ij} = -A_{ji}$, $B_{ij} = -B_{ji}$, $C_{ij} = C_{ji}$, and $D_{ij} = D_{ji}$, we have the inequalities

$$A_{ij}\mu^2+B_{ij}\mu+C_{ij}+D_{ij}\geq 0 \text{ and } -A_{ij}\mu^2-B_{ij}\mu+C_{ij}-D_{ij}\geq 0. \tag{12c}$$

The second inequality is obtained by interchanging i and j and using the symmetry or antisymmetry of the coefficients. Only the cases where $i \neq j$ need be considered, so there are $n(n-1)/2$ such pairs of equations, where n is the number of degrees of freedom of the robot.

If $A_{ij} = B_{ij} = 0$, we have $C_{ij} - D_{ij} \geq 0$ and $C_{ij} + D_{ij} \geq 0$, which are always true if the robot is "strong" enough so that it can stop and hold its position at all points on the desired path. If $A_{ij} = 0$ and $B_{ij} \neq 0$, then we have a pair of linear inequalities which determine a closed interval for μ. If $A_{ij} \neq 0$, then, without loss of generality, we can assume that $A_{ij} > 0$. Then the left-hand side of the first of the inequalities (12c) is a parabola which is concave upward, whereas for the second, it is concave downward. When the parabola is concave downward, then the inequality holds when μ is between the two roots of the quadratic. If the parabola is concave upward, then the inequality holds outside of the region between the roots (Fig. 1). Thus, in one case μ must lie within a closed interval and in the other it must lie outside an open interval, unless of course the open interval is of length zero. In that case, the inequality is always satisfied and the roots of the quadratic will be complex.

Since the admissible values of μ are those which satisfy all of the inequalities, the admissible values must lie in the intersection of all the regions determined by the inequalities. There are $n(n-1)/2$ inequalities which give closed intervals, so the intersection of these regions is also a closed interval. The other $n(n-1)/2$ inequalities, when intersected with this closed interval, each may have the effect of "punching a hole" in the interval (Fig. 2). It is thus possible to have, for any particular value of λ, a set of admissible values for μ which consists of as many as $n(n-1)/2 + 1$ distinct intervals. When the phase portrait of the optimal path is drawn, it may be necessary to have the optimal trajectory dodge the little "islands" which can occur in the admissible region of the phase plane. (Hereafter, these inadmissible regions will be referred to as *islands of inadmissibility*, or just *islands*.) It should be noted, though, that if there is no friction, then $B_{ij} = 0$, which means that in the concave upward case the inequality is satisfied for all values of μ. Thus, in this case there will be no islands in the admissible region.

In addition to the constraints on μ described above, we must also have $\mu \geq 0$. This can be shown as follows: if $\mu < 0$, then the trajectory has passed below the line $\mu = 0$. Below this line, the trajectories always move to the left, since $\mu \equiv d\lambda/dt < 0$. Since the optimal trajectory must approach the desired final state through positive values of μ, the trajectory would then have to pass through $\mu = 0$ again, and would pass from $\mu < 0$ to $\mu > 0$ at a point to the left of where it had passed from $\mu > 0$ to $\mu < 0$.

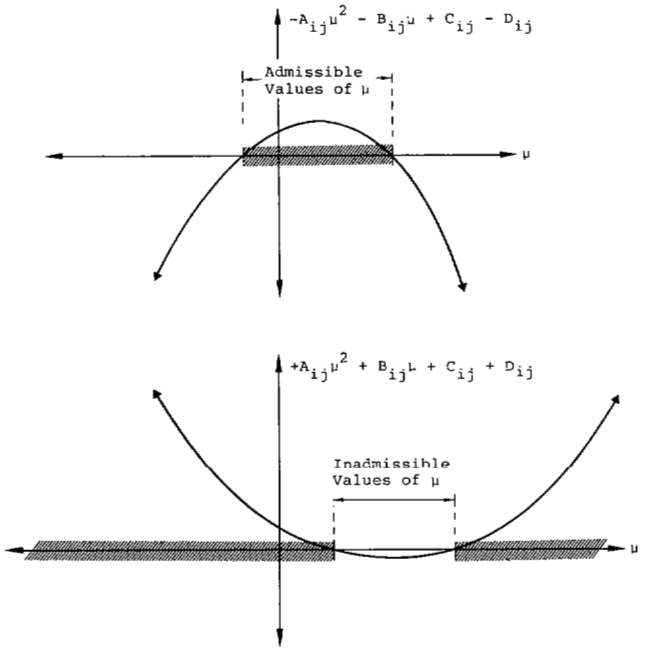


Fig. 1. Admissible regions of μ determined by a pair of parabolic constraints.

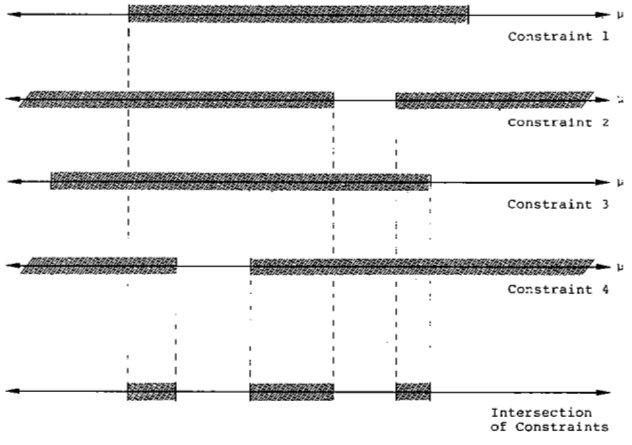


Fig. 2. Intersection of admissible regions of μ.

Thus, in order to get to the desired final state, the trajectory would have to cross itself, forming a loop. But, then, there is no sense in traversing the loop; it would take less time to just use the crossing point as a switching point. Thus, we need consider only those points of the phase plane for which $\mu \geq 0$.

Another way of thinking about the system phase portrait is to assign a pair of vectors to each point in the phase plane. One vector represents the slope when the system is accelerating (i.e., $\dot{\mu}$ is maximized) and the other represents the slope for deceleration (i.e., $\dot{\mu}$ is minimized). This pair of vectors looks like a pair of scissors, and as the position in the phase plane changes, the angles of both the upper and lower jaws of the pair of scissors change. In particular, the angle between the two vectors varies with position. The phase trajectories must, at every point of the phase plane, point in a direction which lies between the jaws of the scissors. At particular points of the phase plane, though, the jaws of the scissors close completely, allowing only a single value for the slope. At other points the scissors may try to go past the closed position, allowing no trajectory at all. This phenomenon, and the condition $\mu \geq 0$, determine the admissible region of the phase plane. This is illustrated in Fig. 3. Note that the boundary of the admissible region passes through those points which have only a single vector associated with them, corresponding to those states where only a single acceleration value is permitted.

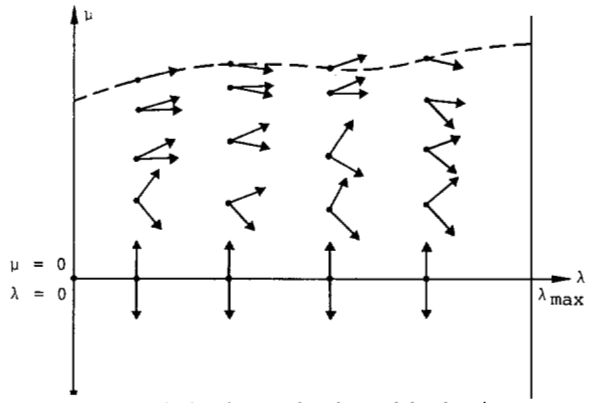


Fig. 3. Phase portrait showing acceleration and deceleration vectors at each state with $\mu_0 = 0$.

IV. Determination of Optimal Trajectories

For illustrative puposes, we first present an algorithm for finding the optimal trajectories for which there are no islands in the phase plane which need to be dodged. The only restrictions, then, will be that μ must lie between a pair of values which are easily calculable, given λ. The optimal trajectory can be constructed by the following steps called the algorithm for constructing optimal trajectories, no islands (ACOTNI).

Step 1: Start at $\lambda = 0$, $\mu = \mu_0$ and construct a trajectory that has the maximum acceleration value. Continue this curve until it either leaves the admissible region of the phase plane or goes past $\lambda = \lambda_{max}$. Note that "leaves the admissible region" implies that if part of the trajectory happens to coincide with a section of the admissible region's boundary, then the trajectory should be extended along the boundary. It is not sufficient in this case to continue the trajectory only until it touches the edge of the admissible region.

Step 2: Construct a second trajectory that starts at $\lambda = \lambda_{max}$, $\mu = \mu_f$ and proceeds *backwards*, so that it is a decelerating curve. This curve should be extended until it either leaves the admissible region or extends past $\lambda = 0$.

Step 3: If the two trajectories intersect, then stop. The point at which the trajectories intersect is the (single) switching point, and the optimal trajectory consists of the first (accelerating) curve from $\lambda = 0$ to the switching point, and the second (decelerating) curve from the switching point to $\lambda = \lambda_{max}$ (Fig. 4).

Step 4: If the two curves under consideration do not intersect, then they must both leave the admissible region. Call the point where the accelerating curve leaves the admissible region λ_1. This is a point on the boundary curve of the admissible region (Fig. 5). If the boundary curve is given by $\mu = g(\lambda)$, then search along the curve, starting at λ_1, until a point is found at which the quantity $\phi(\lambda) \equiv d\mu/d\lambda - dg/d\lambda$ changes sign. (Note that since $g(\lambda)$ determines the boundary of the admissible region, there is only one allowable value of $d\mu/d\lambda$. Also note that if $g(\lambda)$ has a discontinuity, $dg/d\lambda$ must be treated as $+\infty$ or $-\infty$ depending upon the direction of the jump.) This point is the next switching point. Call it λ_d.

Step 5: Construct a decelerating trajectory backwards from λ_d until it intersects an accelerating trajectory. This gives another switching point (see point A in Fig. 6).

Step 6: Construct an accelerating trajectory starting from λ_d. Continue the trajectory until it either intersects the final decelerating trajectory or it leaves the admissible region. If it intersects the decelerating trajectory, then the intersection gives another switching point (see point C in Fig. 6), and the procedure terminates. If the trajectory leaves the admissible region, then go to Step 4.

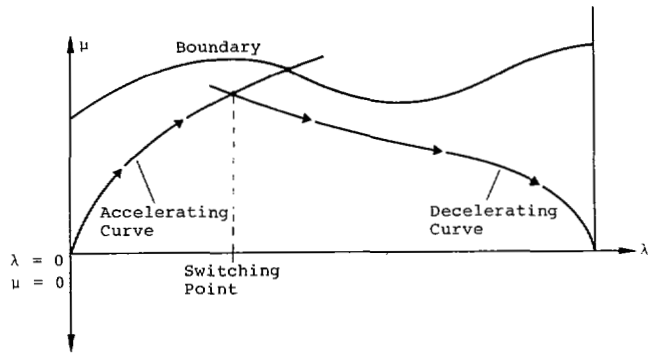


Fig. 4. Case when accelerating and decelerating curves intersect with $\mu_0 = 0$.

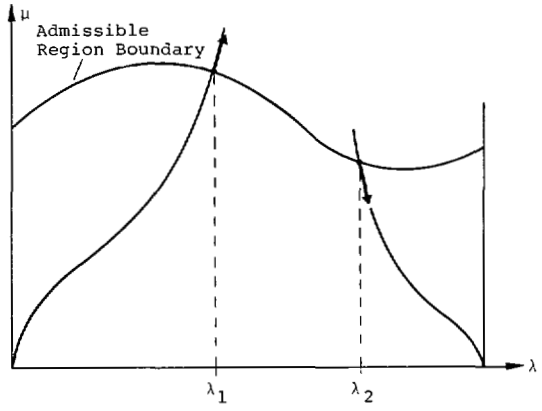
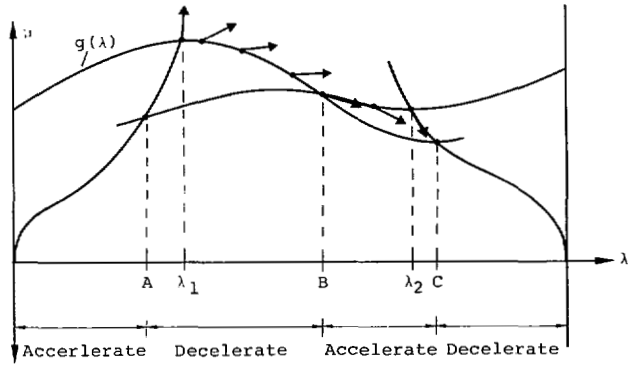


Fig. 5. Case when accelerating and decelerating curves do not intersect.



A, B, and C are switching points
B is a point of osculation between g(λ) and the trajectory

Fig. 6. Complete optimal trajectory formed by ACOTNI with three switching points.

This algorithm yields a sequence of alternately accelerating and decelerating curves which gives the optimal trajectory. Before discussing the optimality of the trajectory, one has to show that all steps of the ACOTNI are possible and that the ACOTNI will terminate.

Addressing the first question, Steps 1-3, 5, and 6 are clearly possible. Step 4 requires finding a sign change of the function $\phi(\lambda)$. Since $\phi(\lambda)$ must be greater than zero where the accelerating trajectory leaves the admissible region and less than zero where the decelerating trajectory leaves, there must be a sign change. Therefore, all steps are possible.

In order to prove that ACOTNI terminates, we must make some assumptions about the form of the functions $f(\lambda)$. In particular, it will be assumed that the f^i-are *piecewise analytic* and are composed of a finite number of pieces in addition to being real-valued. Under these assumptions, the following theorem proves the convergence of ACOTNI within a finite number of iterations.

Theorem 1: If the functions f^i are composed of a finite number of analytic, real-valued pieces, then the function $\phi(\lambda)$ has a finite number of intervals over which it is identically zero and a finite number of zeros outside those intervals.

Proof: The inertia matrix, Coriolis array, and gravitational loading vector are all piecewise analytic in the q^i, and since the $f(\lambda)$ are analytic in λ, the inertia matrix, etc., when expressed as functions of λ [as in (4a) and (4b)] are piecewise analytic and have a finite number of analytic pieces. The functions M_i, Q_i, R_i, S_i of (7b) are, therefore, also piecewise analytic. Since a real-valued analytic function with no singularities in a finite interval must either have a finite number of zeros in that interval or be identically zero, the quantities M_i must either be identically zero in the interval considered or have a finite number of zeros. We cannot have all of the M_i zero, for if that were the case we would have $J_{ij}df^i/d\lambda df^j/d\lambda = M_i df^i/d\lambda = 0$, which is not allowed by hypothesis. If only one of the M_i is nonzero, then there is no boundary curve to deal with, and so no zeros. With two or more not identically zero, there will be a boundary curve. The curve is given by one of the equations (12c) (with "$\geq$" replaced by "$=$") for some pair of indexes i and j. Since the coefficients A, B, C, and D in (12c) are analytic except at the zeros of the M_i, and because the M_i have a finite number of zeros, we can divide the interval under consideration further, using the zeros of the M_i as division points. Within each subinterval, then, only one of the equations (12c) holds. Since (12c) determines μ as an analytic function of λ within this interval, the bounding curve $g(\lambda)$ is piecewise analytic. The curve $\phi(\lambda)$, then, is also piecewise analytic and is either identically zero or has a finite number of zeros in each subinterval. Thus, since $\phi(\lambda)$ either is identically zero in each subinterval or has a finite number of zeros in the subinterval, the number of subintervals is finite, and the number of intervals is finite, the number of zeros and zero-intervals is finite. Q.E.D.

Finally, the following theorem proves the optimality of the solution generated by the ACOTNI.

Theorem 4: Any trajectory generated by the ACOTNI is optimal in the sense of minimum-time control.

Proof: Proof of this theorem is straightforward. Let Γ be the trajectory generated by ACONTI, and let Γ be a trajectory with a shorter traversal time. Now observe three facts. 1) From the form (11) of the cost T, there must be a point (λ_0, μ') on Γ' which is higher than the point (λ_0, μ) on Γ, i.e., $\mu' > \mu$. Otherwise, we would not have a trajectory with a smaller travel time. 2) The trajectory Γ consists of alternately accelerating and decelerating segments, and can therefore be divided into sections which consist of one accelerating and one decelerating segment. 3) The admissible portions of these sections which lie above Γ are bounded on the left and right by either the line $\lambda = 0$, the line $\lambda = \lambda_{max}$, the boundary of the admissible region, or the ACOTNI trajectory itself. Now consider the point (λ, μ') and the trajectory Γ'. This trajectory, if extended backward and forward from (λ_0, μ') must intersect a single section of the ACOTNI trajectory at two or more points, since otherwise it would either leave the admissible region or not meet the initial or final boundary conditions. One such point must occur for $\lambda < \lambda_0$ and one must occur for $\lambda > \lambda_0$. But since the accelerating segment of the trajectory precedes the decelerating segment, the new trajectory must either intersect the accelerating part of the ACOTNI trajectory twice, intersect the decelerating part twice, or first intersect the accelerating part then the decelerating part. But since the torques were chosen so as to minimize or maximize U in (10), any of these situations leads to a contradiction of a theorem on differential inequalities presented in [7]. Q.E.D.

The whole idea of the algorithm is to generate trajectories which come as close as possible to the edge of the admissible

region without actually passing outside it. Thus, the trajectories just barely touch the inadmissible region. In practice this would, of course, be highly dangerous, since minute errors in the control inputs or measured system parameters would very likely make the robot stray from the desired path. Theoretically, however, this trajectory is the minimum-time optimum.

We are now in a position to consider the general case, i.e., the case in which friction, copper losses in the drive motor, etc., are sufficient to cause islands in the phase plane. In this case, the algorithm is most easily presented in a slightly different form. Since there may be several boundary curves instead of one, it is not possible to search a single function for zeros, as was done in ACOTNI. Thus, instead of looking for zeros as the algorithm progresses, we look for them all at once instead, and then construct the trajectories which "just miss" the boundaries, whether the boundaries be the edges of the admissible region or the edges of islands. The appropriate trajectories can then be found by searching the resulting directed graph, always taking the highest trajectory possible, and backtracking when necessary. More formally, the algorithm for construction of optimal trajectories (ACOT) is as follows.

Step 1: Construct the initial accelerating trajectory (same as ACOTNI).

Step 2: Construct the final decelerating trajectory (same as ACOTNI).

Step 3: Calculate the function $\phi(\lambda)$ for the edge of the admissible region and for the edges of all the islands. At each of the sign changes of $\phi(\lambda)$, construct a trajectory for which the sign change is a switching point, as in ACOTNI Steps 5 and 6. The switching direction (acceleration-to-deceleration or vice-versa) should be chosen so that the trajectory does not leave the admissible region. Extend each trajectory until it either leaves the admissible region, or goes past λ_{max}.

Step 4: Find all the intersections of the trajectories. These are potential switching points.

Step 5: Starting at $\lambda = 0$, $\mu = \mu_0$, traverse the grid formed by the various trajectories in such a way that the highest trajectory from the initial to the final points is followed. This is described below in the *grid traversal algorithm* (GTA). Traversing the grid formed by the trajectories generated in Steps 3 and 4 above is a search of a directed graph, where the goal to be searched for is the final decelerating trajectory. If one imagines searching the grid by walking along the trajectories, then one would try to keep making left turns, if possible. If a particular turn led to a dead end, then it would be necessary to backtrack, and take a right turn instead. The whole procedure can best be expressed recursively, in much the same manner as tree traversal procedures.

The algorithm consists of two procedures, one which searches accelerating curves and one which searches decelerating curves. The algorithm is as follows.

AccSearch

On the current (accelerating) trajectory, find the last switching point. At this point, the current trajectory meets a decelerating curve. If that curve is the final decelerating trajectory, then the switching point under consideration is a switching point of the final optimal trajectory. Otherwise, call DecSearch, starting at the current switching point. If DecSearch is successful, then the current point is a switching point of the optimal trajectory. Otherwise, move back along the current accelerating curve to the previous switching point and repeat the process.

DecSearch

On the current (decelerating) trajectory, find the first switching point. Apply AccSearch, starting on this point. If successful, then the current point is a switching point of the optimal trajectory. Otherwise, move forward to the next switching point and repeat the process.

These two algorithms always look first for the curves with the highest velocity, since AccSearch always starts at the end of an accelerating curve and DecSearch always starts at the beginning of a decelerating curve. Therefore, the algorithm finds (if possible) the trajectory with the highest velocity, and hence the smallest traversal time.

The proofs of optimality and convergence of this algorithm are virtually identical to those of ACOTNI, and will not be repeated here. Note that in the convergence proof for ACOTNI the fact that there is only a single boundary curve in the zero-friction case is never used; the same proof therefore applies in the high-friction case.

V. Application Examples

To show how the minimum-time algorithm works, a numerical example follows. The robot used in the example is a simple two-degree-of-freedom robot with one revolute and one prismatic joint, i.e., a robot which moves in polar coordinates. Despite its simplicity, the example robot is sufficient to show the most important aspects of our trajectory planning method. The path chosen is a straight line. Before applying the minimum-time algorithm, we must derive the dynamic equations for the robot. This requires calculation of the inertia matrix, so masses and moments of inertia of the robot must be given.

A drawing of our hypothetical robot is shown in Fig. 7. The robot consists of a rotating fixture with moment of inertia J_θ through which slides a uniformly dense rod of length L_r and mass M_r. The payload has mass M_p and moment of inertia J_p, and its center of mass is at the point (x, y) which is L_p units of length from the end of the sliding rod.

In the examples presented here, the robot will be moved from the point $(1, 1)$ to the point $(1, -1)$. The equation of the curve can be expressed as $r = \sec\theta$, where θ ranges from $+\pi/4$ to $-\pi/4$. Introducing the parameter λ, one possible parameterization is

$$\theta = \frac{\pi}{4} - \lambda \quad r = \sec\left(-\frac{\pi}{4} - \lambda\right). \tag{13}$$

Now introduce the shorthand expressions $M_t \equiv M_r + M_p$, $K \equiv M_r(L_r + 2L_p)$ and $J_t \equiv J_\theta + J_p + M_r(L_p^2 + L_rL_p + L_r^2/3)$. Plugging these expressions and the expressions for the derivatives of r and θ into the dynamic equations gives (see [12] for a detailed derivation)

$$\begin{aligned} u_r = &-M_t \sec\left(\frac{\pi}{4}-\lambda\right)\tan\left(\frac{\pi}{4}-\lambda\right)\dot{\mu} \\ &-k_r \sec\left(\frac{\pi}{4}-\lambda\right)\tan\left(\frac{\pi}{4}-\lambda\right)\mu \\ &+\left[M_t \sec\left(\frac{\pi}{4}-\lambda\right)\right. \\ &\cdot\left(\sec^2\left(\frac{\pi}{4}-\lambda\right)+\tan^2\left(\frac{\pi}{4}-\lambda\right)\right) \\ &\left.+\frac{K}{2}-M_t \sec\left(\frac{\pi}{4}-\lambda\right)\right]\mu^2 \end{aligned} \tag{14a}$$

$$\begin{aligned} u_\theta = &-\left[J_t - K \sec\left(\frac{\pi}{4}-\lambda\right)+M_t \sec^2\left(\frac{\pi}{4}-\lambda\right)\right]\dot{\mu}-k_\theta\mu \\ &+\mu^2\left(2M_t \sec\left(\frac{\pi}{4}-\lambda\right)-K\right)\sec\left(\frac{\pi}{4}-\lambda\right) \\ &\cdot\tan\left(\frac{\pi}{4}-\lambda\right). \end{aligned} \tag{14b}$$

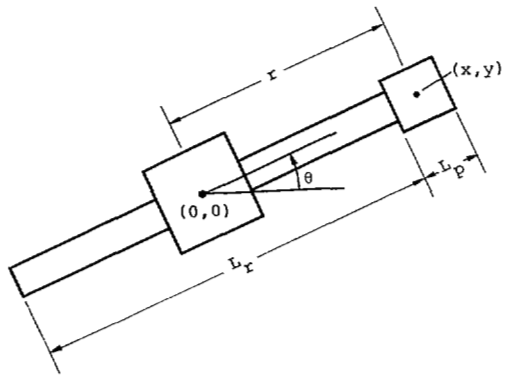


Fig. 7. An example manipulator with extensory and rotational joints.

Solving for $\dot{\mu}$, we have

$$\dot{\mu}=\frac{-1}{M_t \sec\left(\frac{\pi}{4}-\lambda\right)\tan\left(\frac{\pi}{4}-\lambda\right)}\left[u_r+k_r\mu\sec\left(\frac{\pi}{4}-\lambda\right)\tan\left(\frac{\pi}{4}-\lambda\right)-\mu^2\left\{M_t\sec\left(\frac{\pi}{4}-\lambda\right)\left(\sec^2\left(\frac{\pi}{4}-\lambda\right)+\tan^2\left(\frac{\pi}{4}-\lambda\right)\right)+\frac{K}{2}-M_t\sec\left(\frac{\pi}{4}-\lambda\right)\right\}\right] \tag{15a}$$

and

$$\dot{\mu}=\frac{u_\theta+k_\theta\mu-\mu^2\left\{2M_t\sec\left(\frac{\pi}{4}-\lambda\right)-K\right\}\sec\left(\frac{\pi}{4}-\lambda\right)\tan\left(\frac{\pi}{4}-\lambda\right)}{J_t+K\sec\left(\frac{\pi}{4}-\lambda\right)+M_t\sec^2\left(\frac{\pi}{4}-\lambda\right)}. \tag{15b}$$

The signs of the coefficients of u_r and u_θ are

$$\operatorname{sgn}(u_r)=\begin{cases}-1 & 0<\lambda<\frac{\pi}{4}\\ +1 & \frac{\pi}{4}<\lambda<\frac{\pi}{2}\end{cases} \text{ and } \operatorname{sgn}(u_\theta)=1.$$

The limits on $\dot{\mu}$ imposed by the θ joint are the same over the whole interval. For simplicity, let $u^i_{\max}=-u^i_{\max}$ for $i=r,\theta$, then the limits are

$$\dot{\mu}\le\frac{u^\theta_{\max}+\left[\left\{2M_t\sec\left(\frac{\pi}{4}-\lambda\right)-K\right\}\sec\left(\frac{\pi}{4}-\lambda\right)\tan\left(\frac{\pi}{4}-\lambda\right)\right]\mu^2-k_\theta\mu}{J_t-K\sec\left(\frac{\pi}{4}-\lambda\right)+M_t\sec^2\left(\frac{\pi}{4}-\lambda\right)} \tag{16a}$$

and

$$\dot{\mu}\le\frac{-u^\theta_{\max}+\left[\left\{2M_t\sec\left(\frac{\pi}{4}-\lambda\right)-K\right\}\sec\left(\frac{\pi}{4}-\lambda\right)\tan\left(\frac{\pi}{4}-\lambda\right)\right]\mu^2-k_\theta\mu}{J_t-K\sec\left(\frac{\pi}{4}-\lambda\right)+M_t\sec^2\left(\frac{\pi}{4}-\lambda\right)}. \tag{16b}$$

For the r joint, consider the case when $\lambda<\pi/4$. Then we also have

$$\dot{\mu}\le\frac{u^r_{\max}+\left[2M_t\sec\left(\frac{\pi}{4}-\lambda\right)\tan^2\left(\frac{\pi}{4}-\lambda\right)+\frac{K}{2}\right]\mu^2-k_r\mu\sec\left(\frac{\pi}{4}-\lambda\right)\tan\left(\frac{\pi}{4}-\lambda\right)}{M_t\sec\left(\frac{\pi}{4}-\lambda\right)\tan\left(\frac{\pi}{4}-\lambda\right)} \tag{16c}$$

and

$$\dot{\mu} \leq \frac{-u^r_{\max} + \left[2M_t \sec\left(\frac{\pi}{4}-\lambda\right)\tan^2\left(\frac{\pi}{4}-\lambda\right)+\frac{K}{2}\right]\mu^2 - k_r\mu \sec\left(\frac{\pi}{4}-\lambda\right)\tan\left(\frac{\pi}{4}-\lambda\right)}{M_t \sec\left(\frac{\pi}{4}-\lambda\right)\tan\left(\frac{\pi}{4}-\lambda\right)}. \tag{16d}$$

For $\lambda > \pi/4$ the limits have the signs of $u^r_{\max}$ reversed.

Equating upper and lower limits on $\dot{\mu}$ gives the boundary of the admissible region. For $\lambda < \pi/4$, (16b) and (16c) give

$$A\mu^2 + B\mu + C \geq 0$$

where

$$A = -KM_t \sec^4\left(\frac{\pi}{4}-\lambda\right) + 2M_t \sec^3\left(\frac{\pi}{4}-\lambda\right) + \frac{3}{2}KM_t \sec^2\left(\frac{\pi}{4}-\lambda\right) - \left(2M_tJ_t + \frac{K^2}{2}\right)\sec\left(\frac{\pi}{2}-\lambda\right) + \frac{KJ_t}{2}$$

$$B = (J_tk_r - M_tk_\theta)\sec\left(\frac{\pi}{4}-\lambda\right)\tan\left(\frac{\pi}{4}-\lambda\right) - Kk_r \sec^2\left(\frac{\pi}{4}-\lambda\right) \cdot \tan\left(\frac{\pi}{4}-\lambda\right) + M_tk_r \tan\left(\frac{\pi}{4}-\lambda\right)\sec^3\left(\frac{\pi}{4}-\lambda\right)$$

$$C = u^r_{\max}\left(J_t - K\sec\left(\frac{\pi}{4}-\lambda\right) + M_t \sec^2\left(\frac{\pi}{4}-\lambda\right)\right) + u^\theta_{\max} M_t \tan\left(\frac{\pi}{4}-\lambda\right)\sec\left(\frac{\pi}{4}-\lambda\right).$$

Likewise, (16a) and (16d) give

$$-A\mu^2 - B\mu + C \geq 0.$$

The same inequalities, with $u^r_{\max}$ negated, work when $\lambda \geq \pi/4$.

Finally, we need to determine the differential equations to be solved. These equations are

$$\dot{\mu} = \frac{1}{J_t - K\sec\left(\frac{\pi}{4}-\lambda\right) + M_t \sec^4\left(\frac{\pi}{4}-\lambda\right)} \cdot \left[-u_\theta - u_r \sec\left(\frac{\pi}{4}-\lambda\right)\tan\left(\frac{\pi}{4}-\lambda\right) + 2\mu^2 M_t \sec^4\left(\frac{\pi}{4}-\lambda\right)\tan\left(\frac{\pi}{4}-\lambda\right) - \left\{k_r \tan^2\left(\frac{\pi}{4}-\lambda\right)\sec^2\left(\frac{\pi}{4}-\lambda\right) + k_\theta\right\}\mu\right] \tag{17a}$$

$$\lambda = \mu. \tag{17b}$$

The numerical values of the various constants which describe the robot are given in Table I. Using these data, the differential equations were solved numerically using the fourth-order Runge–Kutta method, the program being written in *C* and run under the UNIX[3] operating system on a VAX-11/780.[4] The derivative of the boundary curve $g(\lambda)$ [needed to compute the function $\phi(\lambda)$] was calculated numerically, and the sign changes of $\phi(\lambda)$ found by bisection. The graphs of the resulting trajectories and of the boundary of the admissible region are given in Fig. 8 for the zero-friction case and in Figs. 9 and 10 for the high-friction case.

Note in particular the shape of the admissible region boundary in Fig. 9. For values of λ less than about 0.42 there is not a single range of admissible velocities, but two ranges. Thus, there is an "island" in the phase plane, although the island is chopped off by the constraint that λ be positive. While the existence of such islands may at first seem to defy intuition, the example shows that they do indeed exist. In this case, the island does not really come into play in the calculation of the optimal trajectory. Nevertheless, the example does demonstrate that there may be situations where the admissible region has a fairly complicated shape. Since most practical manipulators have more than two joints and have more complicated dynamic equations than those of the simple robot used here, it is conceivable that the admissible region of the phase plane for a practical robot arm could have quite a complicated shape.

As a final example, to demonstrate clearly the existence of islands in the phase plane, we include a sketch of the admissible region of the phase plane for a two-dimensional Cartesian robot moving along a circular path. In this case, the dynamic equations are a simple pair of uncoupled, linear differential equations with constant coefficients, i.e., $u_x = m\ddot{x} + k_x\dot{x}$, $u_y = m\ddot{y} + k_y\dot{y}$ where $m \equiv$ mass of x and y joints, $k_x \equiv$ coefficient of friction of x joint, and $k_y \equiv$ coefficient of friction of y joint.

Moving this manipulator in a unit circle, say in the first quadrant, requires that

$$x = \cos\lambda,\ y = \sin\lambda,\ 0 \leq \lambda \leq \frac{\pi}{2}.$$

Plugging these expressions and their derivatives into the dynamic equations gives

$$u_x = -m\dot{\mu}\sin\lambda - m\mu^2\cos\lambda - k_x\mu\sin\lambda$$

$$u_y = -m\dot{\mu}\cos\lambda - m\mu^2\sin\lambda - k_y\mu\cos\lambda.$$

Now let the torque bounds be $-T \leq u_x, u_y \leq +T$. Then the bounds on $\dot{\mu}$ are

$$\frac{-T - m\mu^2\cos\lambda - k_x\mu\sin\lambda}{m\sin\lambda} \leq \dot{\mu} \leq \frac{+T - m\mu^2\cos\lambda - k_x\mu\sin\lambda}{m\sin\lambda}$$

and

$$\frac{-T + m\mu^2\sin\lambda - k_y\mu\cos\lambda}{m\cos\lambda} \leq \dot{\mu} \leq \frac{+T + m\mu^2\sin\lambda - k_y\mu\cos\lambda}{m\cos\lambda}.$$

[3] UNIX is a trademark of Bell Laboratories.

[4] VAX is a trademark of the Digital Equipment Corporation.

TABLE I
DATA FOR THE EXAMPLE ROBOT

Constant	Description	Value
J_t	Moment of inertia of θ joint	10^{-3} Kg-M
M_r	Mass of sliding rod	4.0 Kg.
L_r	Length of rod	2.0 M.
J_p	Moment of inertia of payload	10^{-6} $Kg.$-M
M_p	Payload mass	1.0 Kg.
L_p	Length of payload	0.1
u_{max}^r	Maximum force on r joint	1.0 $Kg.$-M/sec^2
u_{max}^t	Maximum torque on θ joint	1.0 $Kg.$-M^2/sec^2
k_r	Friction coefficient of r joint	0.0 (low friction)
	Friction coefficient of r joint	15.0 (high friction)
k_t	Friction coefficient of θ joint	0.0

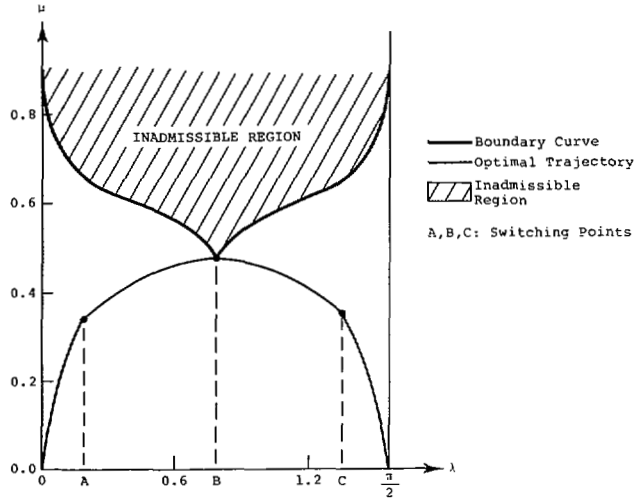


Fig. 8. Optimal trajectory and inadmissible region in case of no island.

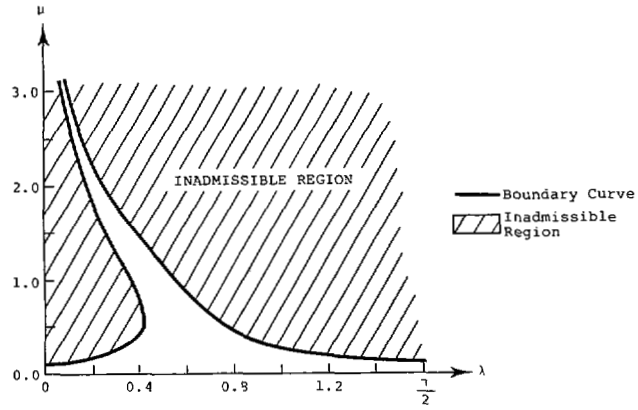


Fig. 9. Inadmissible region in high-friction case.

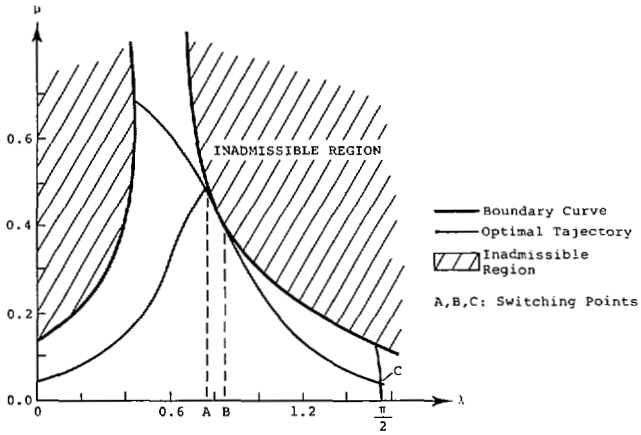


Fig. 10. Optimal trajectory in high-friction case with expanded view of Fig. 9.

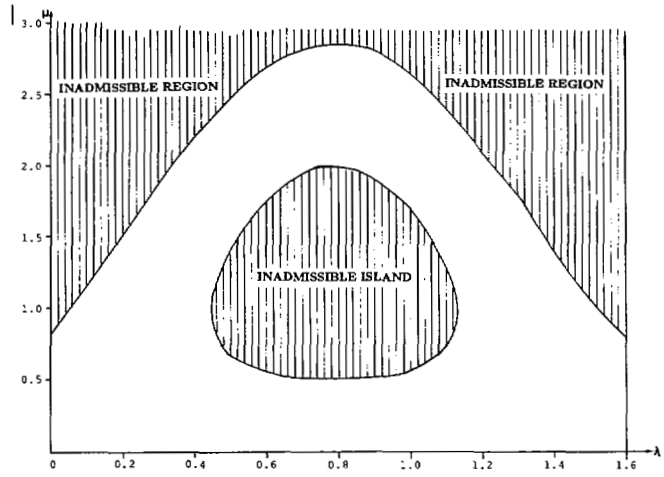


Fig. 11. Example of feasible region with an inadmissible island.

The admissible region consists of the region where the inequalities given above allow some value of the acceleration $\dot{\mu}$, as previously described. Simplifying the resulting inequalities gives the admissible region as that area of the phase plane where

$$m\mu^2+(k_x-k_y)\mu \sin \lambda \cos \lambda+T(\sin \lambda+\cos L)\geq 0$$

and

$$-m\mu^2+(k_x-k_y)\mu \sin \lambda \cos \lambda+T(\sin \lambda+\cos L)\geq 0.$$

Using the values $m = 2$, $k_x = 0$, $k_y = 10$, and $T = \sqrt{2}$ gives the region plotted in Fig. 11 and clearly shows the island.

VI. Discussion and Conclusion

In this paper we have presented a method for obtaining trajectories for minimum-time control of a mechanical arm given the desired geometric path and input torque constraints.

As was already pointed out, the optimal trajectory may actually touch the boundary of the admissible region, generating a rather dangerous case. However, if slightly conservative torque bounds are used in the calculations, then the actual admissible region will be slightly larger than the calculated admissible region, giving some margin for error.

The algorithm has been presented for both the case in which there are no islands in the phase plane and that in which islands do occur. In both cases, the algorithms produce trajectories which "just miss" the inadmissible region, whether the portion of the inadmissible region missed is an island or the region determined by the upper velocity limit. Since the algorithm generates the true minimum-time solution, rather than an approximation to it, the results from the algorithm can provide an absolute reference against which other trajectory planning algorithms can be measured.

Acknowledgment

The authors would like to thank anonymous reviewers for constructive comments and pointing out the missing references [1], [2] in the original version of this paper.

References

[1] J. E. Bobrow, "Optimal control of robotic manipulators," Ph.D. dissertation, Univ. Calif., Los Angeles, CA, Dec. 1982.

[2] J. E. Bobrow, S. Dubowsky, and J. S. Gibson, "On the optimal control robotic manipulators with actuator constraints," in *Proc. 1983 Amer. Contr. Conf.*, San Francisco, CA, June 1983, pp. 782–787.

[3] D. Ter Haar, *Elements of Hamiltonian Mechanics*, 2nd ed. New York: 1971, pp. 35–49.

[4] M. E. Kahn and B. E. Roth, "The near minimum-time control of open-loop articulated kinematic chains," *ASME J. DSMC,* vol. 93, pp. 164–172, Sept. 1971.
[5] B. K. Kim and K. G. Shin, "Suboptimal control of industrial manipulators with a weighted minimum time fuel criterion," *IEEE Trans. Automat. Contr.,* vol. AC-30, pp. 1–10, Jan. 1985.
[6] D. E. Kirk, *Optimal Control Theory: An Introduction.* Englewood Cliffs, NJ: Prentice-Hall, 1971, pp. 227–238.
[7] V. Lakshmikantham and S. Leela, *Differential and Integral Inequalities.* New York: Academic, 1969, pp. 41–43.
[8] C.—S. Lin, P.-R. Chang, and J. Y. S. Luh, "Formulation and optimization of cubic polynomial joint trajectories for mechanical manipulators," *IEEE Trans. Automat. Contr.,* vol. AC-28, pp. 1066–1074, Dec. 1983.
[9] J. Y. S. Luh and M. W. Walker, "Minimum-time along the path for a mechanical arm," in *Proc. 16th Conf. Decision Contr.,* Dec. 1977, pp. 755–759.
[10] J. Y. S. Luh, and C. S. Lin, "Optimum path planning for mechanical manipulators," *ASME J. Dynam. Syst., Measurement, Contr.,* vol. 2, pp. 330–335, June 1981.
[11] R. P. C. Paul, *Robot Manipulators: Mathematics, Programming, and Control.* Cambridge, MA: M.I.T. Press, 1981, pp. 157–195.
[12] K. G. Shin and N. D. McKay, "Open-loop minimum-time control of mechanical manipulators and its application," in *Proc. 1984 Amer. Contr. Conf.,* San Diego, CA, June 1984, pp. 1231–1236.

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

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

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

Neil D. McKay was born in Albany, NY, on May 5, 1958. He received the B.S. degree in electrical engineering from the University of Rochester, Rochester, NY, in 1980 and the M.S. degree in 1982 from Rensselaer Polytechnic Institute, Troy, NY.

While at Rensselaer Polytechnic Institute, he was a Rockwell Fellow. He is currently completing the Ph.D. degree in electrical engineering at the University of Michigan, Ann Arbor. His research interests include robot control, robot path planning, and optimal control theory.