AN EFFICIENT ROBOT ARM CONTROL UNDER GEOMETRIC PATH CONSTRAINTS[1]

Kang G. Shin and Neil D. McKay

Department of Electrical and Computer Engineering
The University of Michigan
Ann Arbor, Michigan 48109

ABSTRACT

Conventionally, robot control algorithms are divided into two stages, namely, *path planning* and *path tracking* (or *path control).* This division has been adopted mainly as a means of alleviating difficulties in dealing with complex, coupled robot arm dynamics. Unfortunately, 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 robot arm in minimum time along a specified geometric path subject to input torque/force constraints. We first describe the robot arm dynamics using parametric functions which represent geometric path constraints to be honored for collision avoidance as well as task requirements. Secondly, constraints on input torques/forces are converted to those on the parameters. Finally, the minimum-time solution is deduced in an algorithm form using phase-plane techniques.

1. Introduction

During the past several years a great deal of attention has been focused on industrial automation techniques, especially the use of general-purpose robots. Since the purpose of industrial robots is to increase productivity, an obvious question to ask is how robots should be controlled so as to produce as many units as possible per dollar invested. The usual assumption is that fixed costs dominate the cost per item produced, so that it is desirable to produce as many units as possible in a given time.

There are a variety of algorithms available for minimum-time or near-minimum-time robot arm control. These algorithms usually assume that the control structure of the robot has been divided into two levels. The first level is called *path planning*, and the second level is called *path control* or *path tracking* The usual definition of path control is the act of attempting to make the robot's actual position and velocity match desired values of position and velocity; the desired values are provided to the controller by the path planner. The path planner receives as input some sort of spatial path descriptor from which it calculates a time history of the desired positions and velocities. The path tracker then takes care of any deviations of the actual position and velocity from the desired values.

The reason for dividing the control scheme in this way is that 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 robot arm dynamics frequently are not taken into account at this level, such trackers can generally keep the robot arm fairly close to the desired trajectory. More sophisticated methods can be used, though, such as resolved motion rate control[1], resolved acceleration control[2], and various adaptive techniques[3]-[5].

Unfortunately, the simplicity obtained from the division into path planning and path tracking comes at the expense of efficiency. The source of the inefficiency is the path planner. In order to use the robot efficiently, the path 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 path 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 [6,7]. 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 under-utilization of the robot.

To alleviate the inefficiency, this paper presents a solution to the minimum-time robot arm path control problem subject to constraints on its geometric path and input torques/forces. The solution will be in the form of a path planning algorithm, and will take into account the details of the dynamics of the robot arm. The output of the path planner will be the true minimum-time solution, and so will be useful as a standard against which the performance of other path 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 [8,9].

The remainder of this paper is divided into five sections. Section 2 describes a method for making the robot arm dynamic equations more tractable and a method for handling input torque constraints. Section 3 contains a detailed formulation of the minimum-time control problem. In Section 4, the form of the optimal solution is deduced using phase-plane techniques. Section 5 presents the highlight of this paper, that is, an algorithm for generating optimal(i.e. minimum-time) trajectories. The final section is a discussion of the significance of the results.

2. Robot Dynamics with Constraints

Before delving headlong into the problem of minimum-time control, consider the behavior of the system to be controlled, that is, a dynamic model of the robot arm. 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 two most common methods are the Lagrange method and the Newton-Euler method. The Newton-Euler method is computationally efficient, but is a recursive formulation which is hard to deal with in control problems. The Lagrange formulation, while not computationally efficient, does yield a set of differential equations which are easy to manipulate for robot control problems. Since the dynamic equations will be used here only to obtain analytical results, we have used Lagrange's method to derive the following robot arm dynamic equations[12, 13].

[1]*The work reported here is supported in part by the US AFOSR contract No. F49620-82-C-0089 and Robot Systems Division, Center for Robotics and Integrated Manufacturing(CRIM), The University of Michigan, Ann Arbor, Michigan. Any opinions, findings, and conclusions or recommendations in this paper are those of the authors and do not necessarily reflect the view of the funding agencies.*

$$\dot{\mathbf{q}}^i = \mathbf{v}^i \tag{1a}$$

$$\mathbf{u}_i = \mathbf{J}_{ij}(\mathbf{q})\dot{\mathbf{v}}^j + \mathbf{R}_{ij}\mathbf{v}^j + \mathbf{C}_{ijk}(\mathbf{q})\mathbf{v}^j\mathbf{v}^k + \mathbf{G}_i(\mathbf{q}) \tag{1b}$$

where

$\mathbf{q}^i = i^{th}$ generalized coordinate
$\mathbf{v}^i = i^{th}$ generalized velocity
$\mathbf{u}_i = i^{th}$ generalized force
$\mathbf{J}_{ij}$ = the inertia matrix
$\mathbf{G}_i$ = gravitational force on the i^{th} joint
$\mathbf{C}_{ijk}$ = Coriolis force array
$\mathbf{R}_{ij}$ = viscous friction matrix

The Einstein summation convention has been used, and all indices run from one to n inclusive for an n-degree-of-freedom robot.

The elements of the inertia matrix $\mathbf{J}_{ij}$ are constants of proportionality which relate the torque/force exerted on the i^{th} joint to the acceleration of the j^{th} joint. The entries $\mathbf{C}_{ijk}$ of the Coriolis array describe the force felt at joint i due to the velocities of joints j and k. The viscous friction matrix $\mathbf{R}_{ij}$ gives the frictional force felt at joint i due to the velocity at joint j. Note that this matrix is diagonal, and all the entries are non-negative.

The motion of the robot arm will not, of course, be completely unconstrained. In fact, it will later be assumed that the robot arm must be constrained to a fixed path in joint space,[2] 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

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

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

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[10] for an example).

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

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

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

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

$$\mathbf{u}_i = \mathbf{J}_{ij}(\lambda)\frac{df^j}{d\lambda}\dot{\mu} + \mathbf{J}_{ij}(\lambda)\frac{d^2f^j}{d\lambda^2}\mu^2 + \mathbf{G}_i(\lambda) + \mathbf{R}_{ij}\frac{df^j}{d\lambda}\mu + \mathbf{C}_{ijk}(\lambda)\frac{df^j}{d\lambda}\frac{df^k}{d\lambda}\mu^2 \tag{4b}$$

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, a better way of obtaining a single state equation from the n equations given is to multiply the i^{th} equation by $\frac{df^i(\lambda)}{d\lambda}$ and sum over i, giving

$$\mathbf{u}_i\frac{df^i}{d\lambda} = \mathbf{J}_{ij}(\lambda)\frac{df^i}{d\lambda}\frac{df^j}{d\lambda}\dot{\mu} + \mathbf{J}_{ij}(\lambda)\frac{df^i}{d\lambda}\frac{d^2f^j}{d\lambda^2}\mu^2 + \mathbf{G}_i(\lambda)\frac{df^i}{d\lambda} + \mathbf{R}_{ij}\frac{df^i}{d\lambda}\frac{df^j}{d\lambda}\mu + \mathbf{C}_{ijk}(\lambda)\frac{df^i}{d\lambda}\frac{df^j}{d\lambda}\frac{df^k}{d\lambda}\mu^2 \tag{5}$$

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[3] 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 non-zero, 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

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

$$\dot{\mu} = \frac{1}{\mathbf{J}_{ij}(\lambda)\frac{df^i}{d\lambda}\frac{df^j}{d\lambda}}\Big[\mathbf{u}_i\frac{df^i}{d\lambda} - \mathbf{J}_{ij}(\lambda)\frac{df^i}{d\lambda}\frac{d^2f^j}{d\lambda^2}\mu^2 - \mathbf{G}_i(\lambda)\frac{df^i}{d\lambda} - \mathbf{R}_{ij}\frac{df^i}{d\lambda}\frac{df^j}{d\lambda}\mu - \mathbf{C}_{ijk}(\lambda)\frac{df^i}{d\lambda}\frac{df^j}{d\lambda}\frac{df^k}{d\lambda}\mu^2\Big] \tag{6b}$$

Consider now the constraints on the inputs, namely $|\mathbf{u}_i| \le \mathbf{u}^i_{\max}$ and (4b). The dynamic equation (4b) can be viewed as having the following form: $\mathbf{u}_i = \mathbf{g}_i(\lambda)\dot{\mu} + \mathbf{h}_i(\lambda,\mu)$. 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 pseudo-acceleration $\dot{\mu}$ is that all the positions, velocities, and accelerations of the various joints are related to one another through the parameterization of the path, and so the joints in some sense impose restrictions on one another. 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, we can use $-\mathbf{u}^i_{\max} \le \mathbf{u}_i \le +\mathbf{u}^i_{\max}$ so that

$$-\mathbf{u}^i_{\max} \le \mathbf{J}_{ij}\frac{df^j}{d\lambda}\dot{\mu} + \left[\mathbf{J}_{ij}\frac{d^2f^j}{d\lambda^2} + \mathbf{C}_{ijk}\frac{df^j}{d\lambda}\frac{df^k}{d\lambda}\right]\mu^2 + \mathbf{R}_{ij}\frac{df^j}{d\lambda}\mu + \mathbf{G}_i \le +\mathbf{u}^i_{\max} \tag{7a}$$

Introducing some shorthand notation, let

$$M_i \equiv \mathbf{J}_{ij}\frac{df^j}{d\lambda}$$

$$Q_i \equiv \mathbf{J}_{ij}\frac{d^2f^j}{d\lambda^2} + \mathbf{C}_{ijk}\frac{df^j}{d\lambda}\frac{df^k}{d\lambda}$$

$$R_i \equiv \mathbf{R}_{ij}\frac{df^j}{d\lambda}$$

$$S_i \equiv \mathbf{G}_i$$

We then have

[2] *A path is normally described in Cartesian space and the conversion from Cartesian space to joint space is needed here. Since any detailed treatment of the conversion is out of the scope of this paper, only brief remarks on this are given in the next paragraph.*

[3] *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.*

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

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 $M_i \neq 0$)

$$\frac{-\mathbf{u}^i_{\max}-sgn(M_i)\left[Q_i\mu^2+R_i\mu+S_i\right]}{|M_i|} \le \dot{\mu} \tag{7c}$$

$$\le \frac{+\mathbf{u}^i_{\max}-sgn(M_i)\left[Q_i\mu^2+R_i\mu+S_i\right]}{|M_i|}$$

Using the additional shorthand abbreviations

$$LB_i \equiv |M_i|^{-1}\left\{-\mathbf{u}^i_{\max}-sgn(M_i)\left[Q_i\mu^2+R_i\mu+S_i\right]\right\}$$

and

$$UB_i \equiv |M_i|^{-1}\left\{+\mathbf{u}^i_{\max}-sgn(M_i)\left[Q_i\mu^2+R_i\mu+S_i\right]\right\}$$

we have $LB_i \le \dot{\mu} \le UB_i$. 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

$$GLB(\lambda,\mu) \le \dot{\mu} \le LUB(\lambda,\mu) \tag{7e}$$

The difference between the path 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 conventional path 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.

3. Formulation of Optimal Control Problem

Now that we have the dynamic equations of the robot arm along the geometric path and the constraints on its inputs, we can address the actual control problem. In controlling a robot arm, it is desirable to minimize the cost of production subject to whatever physical constraints the robot arm dynamics impose. Such problems 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[11]. 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 attempt to apply the maximum principle, but with the intention of obtaining *qualitative results* rather than analytic or numerical solutions. The results will be used later to develop an algorithm for generating the minimum-time trajectory.

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

$$C=\int_0^{t_f} 1 \cdot dt \tag{8}$$

where the final time t_f is left free. The cost function C must be minimized subject to the three sets of constraints given below. The first set is a set of differential equation constraints, namely the dynamic equations of the robot arm(i.e. (6a) and (6b)). The second set is a set of input constraints. Since the robots joint actuators have some upper bound on the torque which they can provide, the inputs to the robot arm are bounded (i.e. $|\mathbf{u}_i| \le \mathbf{u}^i_{\max}$). The third set of constraints is a set of spatial constraints. The robot must get to its desired final position without colliding with any obstacles. It will be assumed here that the desired geometric path of the robot arm has been pre-planned,[4] and is provided to the minimum-time controller in *parametric form*, as described earlier(i.e. Eq. (3)). It will be assumed that the $\mathbf{q}^i$ 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 $\frac{df^i}{d\lambda}$ 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 $\mathbf{x}^* = (\lambda^*,\mu^*)$ and $\mathbf{u}_i^*$ by minimizing (8) subject to (6a), (6b), $|\mathbf{u}_i| \le \mathbf{u}^i_{\max}$, $0 \le \lambda \le \lambda_{\max}$, and the boundary conditions

$$\mu(0)=\mu_0, \quad \mu(t_f)=\mu_f \tag{9a}$$

$$\lambda(0)=0, \quad \lambda(t_f)=\lambda_{\max} \tag{9b}$$

3.1. Application of the Maximum Principle

In order to apply the state constraint $0\le\lambda\le\lambda_{\max}$, it is convenient to add a third state equation. Call the third state ν, and let the third equation have the form

$$\dot{\nu}=\lambda^2 1(-\lambda)+(\lambda_{\max}-\lambda)^2 1(\lambda-\lambda_{\max}) \tag{10}$$

where

$$1(x)=\begin{cases} 1 & \text{if } x\ge 0 \\ 0 & \text{if } x<0 \end{cases}$$

Note that $\dot{\nu}\ge 0$, so that the boundary conditions $\nu(0)=\nu(t_f)=0$ force ν to be identically zero. But the only way for ν to be identically zero is to have $0\le\lambda\le\lambda_{\max}$, thus forcing λ to remain in the desired interval.

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

$$M(\lambda)\equiv \mathbf{J}_{ij}(\lambda)\frac{df^i}{d\lambda}\frac{df^j}{d\lambda}$$

$$U(\lambda)\equiv \mathbf{u}_i\frac{df^i}{d\lambda}$$

$$Q(\lambda)\equiv \mathbf{J}_{ij}(\lambda)\frac{df^i}{d\lambda}\frac{d^2f^j}{d\lambda^2}+\mathbf{C}_{ijk}(\lambda)\frac{df^i}{d\lambda}\frac{df^j}{d\lambda}\frac{df^k}{d\lambda}$$

$$R(\lambda)\equiv \mathbf{R}_{ij}\frac{df^i}{d\lambda}\frac{df^j}{d\lambda}$$

$$S(\lambda)\equiv \mathbf{G}_i(\lambda)\frac{df^i}{d\lambda}$$

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

$$\dot{\lambda}=\mu \tag{11a}$$

$$\dot{\mu}=\frac{1}{M}\left[U-Q\mu^2-R\mu-S\right] \tag{11b}$$

$$\dot{\nu}=\lambda^2 1(\lambda)+(\lambda_{\max}-\lambda)^2 1(\lambda_{\max}-\lambda) \tag{11c}$$

The M term is a quadratic form reminiscent of the expression for the robot arm's kinetic energy. In fact, if the parametric expressions for the $\dot{\mathbf{q}}_i$ are plugged into the formula for kinetic energy, one obtains the expression $K=M\mu^2/2$. The Q term represents the components of the Coriolis and centrifugal forces which act along the path plus the artificial forces of constraint generated by the parameterization. The R term represents frictional components, and S gives the gravitational force along the path. U is the weighted input term.

Under the above setting the problem MTPP can be converted to the following.

Problem MTPP': Find $\mathbf{y}^* = (\lambda^*,\mu^*,\nu^*)$ and U^* by minimizing (8) subject to (11a), (11b), (11c), (7d), (9a), and (9b).

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

For the MTPP' problem the Hamiltonian now becomes

$$H=1+p_1\mu+\frac{p_2}{\mathbf{J}_{ij}(\lambda)\frac{df^i}{d\lambda}\frac{df^j}{d\lambda}}\Bigg[\mathbf{u}_i\frac{df^i}{d\lambda} \tag{12a}$$
$$-\mathbf{J}_{ij}(\lambda)\frac{df^i}{d\lambda}\frac{d^2f^j}{d\lambda^2}\mu^2-\mathbf{G}_i(\lambda)\frac{df^i}{d\lambda}-\mathbf{R}_{ij}\frac{df^i}{d\lambda}\frac{df^j}{d\lambda}\mu$$
$$-\mathbf{C}_{ijk}(\lambda)\frac{df^i}{d\lambda}\frac{df^j}{d\lambda}\frac{df^k}{d\lambda}\mu^2\Bigg]$$
$$+p_3\left[\lambda^2 1(-\lambda)+(\lambda_{\max}-\lambda)^2 1(\lambda-\lambda_{\max})\right]$$

Or using the foregoing substitutions, the Hamiltonian is

$$H=1+p_1\mu+\frac{p_2}{M}\left[U-Q\mu^2-R\mu-S\right] \tag{12b}$$
$$+p_3\left[\lambda^2 1(-\lambda)+(\lambda_{\max}-\lambda)^2 1(\lambda-\lambda_{\max})\right]$$

Differentiating with respect to μ, we obtain

$$-\dot{p}_2=\frac{\partial H}{\partial \mu}=p_1+\frac{p_2}{M}\left[-2Q\mu-R\right] \tag{13a}$$

Differentiating with respect to λ,

$$-\dot{p}_1=\frac{\partial H}{\partial \lambda}=\frac{p_2}{M}\left[\frac{\partial U}{\partial \lambda}-\frac{dQ}{d\lambda}\mu^2-\frac{dR}{d\lambda}\mu-\frac{dS}{d\lambda}\right]$$
$$-\frac{p_2}{M^2}\left[U-Q\mu^2-R\mu-S\right]\frac{dM}{d\lambda}$$
$$+2p_3\left[\lambda 1(-\lambda)-(\lambda_{\max}-\lambda)1(\lambda-\lambda_{\max})\right]$$

$$=-\frac{p_2}{M^2}\Bigg[U\frac{dM}{d\lambda}-M\frac{\partial U}{\partial \lambda}-(Q\frac{dM}{d\lambda}-M\frac{dQ}{d\lambda})\mu^2 \tag{13b}$$
$$-(R\frac{dM}{d\lambda}-M\frac{dR}{d\lambda})\mu-(S\frac{dM}{d\lambda}-M\frac{dS}{d\lambda})\Bigg]$$
$$+2p_3\left[\lambda 1(-\lambda)-(\lambda_{\max}-\lambda)1(\lambda-\lambda_{\max})\right]$$

Finally, differentiating with respect to ν,

$$-\dot{p}_3=\frac{\partial H}{\partial \nu}=0 \tag{13c}$$

Thus if we are to apply the maximum principle to this problem, we must minimize H in (12b). In addition, constraints (11a), (11b), (11c), (9a), (9b), and (7d) must be met, and H must satisfy the boundary condition.

$$H(\mathbf{y}(t_f),\mathbf{p}(t_f),U(t_f))=0 \tag{14}$$

Here, the state vector $\mathbf{y}$ is the vector (λ,μ,ν), and we have a single input term $U(t_f)=\mathbf{u}_i(\lambda_{\max})\frac{df^i}{d\lambda}(\lambda_{\max})$. In (14) we used the fact that H does not explicitly depend upon t. Also one can see $\mathbf{y}(t_f)=(\lambda_{\max},\mu_f,0)^T$ from the boundary conditions (9), and the condition $\nu(t_f)=0$.

Note that the Hamiltonian functional (12b) is linear in U, and U is bounded because of the boundedness of $\mathbf{u}_i$ and $\frac{df^i}{d\lambda}$ in $[0,\lambda_{\max}]$. This implies that the optimal solution for U should follow a bang-bang policy. That is, at any point on the optimal trajectory the quantity U in (12b) must assume its maximum or minimum. The extremum of U can be obtained by maximizing or minimizing itself with respect to the $\mathbf{u}_i$. Since we may write the equality constraints on the $\mathbf{u}_i$ as $\mathbf{u}_i=\mathbf{g}_i(\lambda)\dot{\mu}+\mathbf{h}_i(\lambda,\mu)$, we have

$$U=\mathbf{u}_i\frac{df^i}{d\lambda}=\mathbf{g}_i(\lambda)\frac{df^i}{d\lambda}\dot{\mu}+\mathbf{h}_i(\lambda,\mu)\frac{df^i}{d\lambda}$$

Because of the bang-bang nature of the control U, and because for a given state (λ,μ) the quantity U is linearly related to $\dot{\mu}$ through the above equation, $\dot{\mu}$ will also be bang-bang. Therefore $\dot{\mu}$ will be equal to either $GLB(\lambda,\mu)$ or $LUB(\lambda,\mu)$. Thinking of the three dimensional case again, $\dot{\mu}$ must put the input at one of the points where the line formed by the input equality constraints meets one of the sides of the prism formed by the inequalities. If the $i-th$ joint input is on one of the sides of the constraint box, then it is driven at its maximum capability, while the others in general are not. The slowest joint, then, is pushed as hard as it will go while the others exert the proper amount of force to keep the robot arm on the right path.

The above discussion is valid whenever the coefficient of the input term is not zero, i.e. when p_2 in (13a) is not zero. If p_2 is zero at isolated points only, then the optimal control is determined almost everywhere. On the other hand, if p_2 is zero on some interval, we have the following theorem.

Theorem 1: If p_2 is zero on some interval $[t_1,t_2]$ $(t_1<t_2)$ then p_2 is zero on the interval $[t_1,\infty)$, and a time-optimal solution does not exist.

Proof: Suppose that p_2 is zero on some interval $[t_1,t_2]$. Over this interval, we have $p_2=0$ and $\dot{p}_2=0$. But in order for this to be true, we must also have $p_1=\dot{p}_1=0$ from (13a) and (13b). Since the boundary conditions on ν prevent the coefficient of p_3 from ever being non-zero, this would force p_1 and p_2 to be zero on the interval $[t_1,\infty)$. This gives $H(\mathbf{y}(t),\mathbf{p}(t),U(t))=1$, $t\geq t_1$, which means that the boundary condition $H(\mathbf{y}(t_f),\mu_f,0,\mathbf{p}(t_f),U(t_f))=0$. cannot be met. It follows that there can be at most one singular interval, and that if there is a singular interval, then no solution to the optimal control problem exists. If there is no singular interval, then the minimum-time solution is unique. Q.E.D.

It is also possible to deduce the signs of $p_2(0)$ and $p_2(t_f)$. Thus we have another theorem.

Theorem 2: If $U_{\max}(0)>S(0)>U_{\min}(0)$ then $p_2(0)<0$ and $p_2(t_f)>0$.

Proof: It is known that $0\leq\lambda\leq\lambda_{\max}$, so that at $t=t_f$ we must have $\dot{\mu}\leq 0$. Otherwise, since $\mu(t_f)=0$, we would have to have had $\mu(t)<0$ for some $t<t_f$. The robot arm would then have had to be moving backwards along the path when it reached the destination $\lambda_{\max}$, and would therefore have to have come from some point $\lambda>\lambda_{\max}$. But at t_f we have $\dot{\mu}(t_f)=M^{-1}(U-S)<0$. Since $M>0$, this gives $U-S<0$. Now the value of H at time t_f is 0, so that

$$H(t_f)=1+\frac{p_2(t_f)}{M}\left[U-S\right]=0$$

If $p_2(t_f)\leq 0$, then $H(t_f)>0$, so we must have $p_2(t_f)>0$.

To determine the sign of $p_2(0)$, consider the quantity $\dot{\mu}(0)$. By an argument similar to that given above, $\dot{\mu}(0)>0$. This gives the result $U-S>0$. Since the control is bang-bang, we must have $U=U_{\max}$, since otherwise $U=U_{\min}$ and $U_{\min}-S<0$. But if $U=U_{\max}$, then p_2 must be less than zero. Therefore $p_2(0)<0$. Q.E.D.

One consequence of these theorems is that the number of switching points is odd. If the number of switching points were even, then the sign of $p_2(t_f)$ would be the same as that of $p_2(0)$, since $sgn(p_2(t_f))=(-1)^m sgn(p_2(0))$, where m is the number of sign changes.

4. Phase Plane Interpretation

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 equation (11b) by (11a). This gives

$$\frac{d\mu}{d\lambda}=\frac{d\mu}{dt}\bigg/\frac{d\lambda}{dt}=\frac{\dot{\mu}}{\mu}=\frac{1}{\mu M}\left[U-Q\mu^2-R\mu-S\right] \tag{15}$$

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

$$T=\int dt=\int_0^{\lambda_{\max}}\frac{dt}{d\lambda}d\lambda=\int_0^{\lambda_{\max}}\frac{1}{\mu}d\lambda \tag{16}$$

The idea, then, is to minimize this integral subject to the given constraints. We therefore want to make 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 $\frac{d\mu}{d\lambda}$, one simply divides the limits on $\dot{\mu}$ by μ, since $\frac{d\mu}{d\lambda} = \frac{\dot{\mu}}{\mu}$.

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 plane 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,

$$\frac{u^i_{\max} - sgn(M_i)\left[Q_i\mu^2 + R_i\mu + S_i\right]}{|M_i|} - \frac{-u^j_{\max} - sgn(M_j)\left[Q_j\mu^2 + R_j\mu + S_j\right]}{|M_j|} \geq 0$$

Rearranging this inequality,

$$\left\{u^i_{\max} - sgn(M_i)\left[Q_i\mu^2 + R_i\mu + S_i\right]\right\}|M_j| - \left\{-u^j_{\max} - sgn(M_j)\left[Q_j\mu^2 + R_j\mu + S_j\right]\right\}|M_i| \geq 0$$

or

$$\left[|M_i|sgn(M_j)Q_j - |M_j|sgn(M_i)Q_i\right]\mu^2 + \left[|M_i|sgn(M_j)R_j - |M_j|sgn(M_i)R_i\right]\mu + \left[|M_i|\left[u^j_{\max} + sgn(M_j)S_j\right] + |M_j|\left[u^i_{\max} - sgn(M_i)S_i\right]\right] \geq 0$$

Dividing this equation by $|M_i||M_j|$ gives

$$\left[\frac{Q_j}{M_j} - \frac{Q_i}{M_i}\right]\mu^2 + \left[\frac{R_j}{M_j} - \frac{R_i}{M_i}\right]\mu + \left[\frac{S_j}{M_j} - \frac{S_i}{M_i}\right] + \left[\frac{u^j_{\max}}{|M_j|} + \frac{u^i_{\max}}{|M_i|}\right] \geq 0 \tag{17a}$$

The left side of this inequality is a quadratic in μ, and the inequality is always true for $\mu=0$ if it is true that $|S_i| \leq u^i_{\max}$ for all i. The bounds on μ may then be found from the quadratic formula.

Introducing yet more shorthand notation, let

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

$$C_{ij} \equiv \left[\frac{u^j_{\max}}{|M_j|} + \frac{u^i_{\max}}{|M_i|}\right] \quad D_{ij} \equiv \left[\frac{S_j}{M_j} - \frac{S_i}{M_i}\right]$$

Do not confuse C_{ij} with either the cost function C or the Coriolis array $\mathbf{C}_{ijk}$. The inequality now becomes

$$A_{ij}\mu^2 + B_{ij}\mu + C_{ij} + D_{ij} \geq 0 \tag{17b}$$

Note that, from the definitions, $A_{ij} = -A_{ji}$, $B_{ij} = -B_{ji}$, $C_{ij} = C_{ji}$, and $D_{ij} = -D_{ji}$. Since the inequality must hold for all i and j, i and j can be interchanged and the symmetry or anti-symmetry of the coefficients used to get the inequality

$$-A_{ij}\mu^2 - B_{ij}\mu + C_{ij} - D_{ij} \geq 0 \tag{17c}$$

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 inequality (17b) is a parabola which is concave upward, whereas for (17c) 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(Figure 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 constraint 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(Figure 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 $n(n+1)/2$ 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 $\mu < 0$. 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 the admissible region of the phase plane includes only points 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 Figure 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.

5. Determination of Optimal Trajectories

For illustrative purposes, we first present an algorithm for finding the optimal trajectories for an ideal, frictionless robot arm. Then, the algorithm will be extended to include the general case. In the case of zero friction, we have a set of $n(n-1)/2$ bounds on μ, and each of these relations is symmetric with respect to $\mu=0$. Also, there are no islands in the phase plane which need to be dodged. The only restriction, then, will be that μ must lie between a pair of continuous curves with piecewise continuous derivatives. The optimal trajectory can be constructed by the following steps called the *Algorithm for Constructing Optimal Trajectories, No Friction (ACOTNF).*

S1. 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.

S2. 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 $\lambda=0$.

S3. If the two trajectories intersect, then the point at which the trajectories intersect is the (single) switching point; 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 $\lambda=\lambda_{max}$(Figure 4). The algorithm then terminates.

S4. 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(Figure 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 $\frac{d\mu}{d\lambda}$ is equal to $\frac{dg}{d\lambda}$. (Note that since $g(\lambda)$ determines the boundary of the admissible region, there is only one allowable value of $\frac{d\mu}{d\lambda}$). This point is the next switching point. Call it λ_d.

S5. Construct a decelerating trajectory backwards from λ_d until it intersects an accelerating trajectory. This gives another switching point (see point A in Figure 6).

S6. 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 Figure 6), and the procedure terminates. If the trajectory leaves the admissible region, then go to step 4.

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

Addressing the first question, steps 1, 2, 3, 5, and 6 are clearly possible. Step 4, however, requires finding a zero of a function. Under the given conditions, does the function always have at least one zero? The answer is yes, for the following reason. Note that at $\lambda=\lambda_1$ the trajectory points outward from the admissible region. Similarly, at the point $\lambda=\lambda_2$ where the decelerating curve passes outside the admissible region, the trajectory must point inward. If the slope of the bounding curve of the admissible region is continuous at these points, then we have

$$\left.\frac{d\mu}{d\lambda}\right|_{\lambda=\lambda_1} > \left.\frac{dg(\lambda)}{d\lambda}\right|_{\lambda=\lambda_1} \quad \text{and} \quad \left.\frac{d\mu}{d\lambda}\right|_{\lambda=\lambda_2} < \left.\frac{dg(\lambda)}{d\lambda}\right|_{\lambda=\lambda_2}$$

where $g(\lambda)$ is the equation of the bounding curve of the admissible region. The quantity $\frac{d\mu}{d\lambda}-\frac{dg(\lambda)}{d\lambda}$ must, therefore, change sign between λ_1 and λ_2. If $g(\lambda)$ has a continuous derivative in this range, then there will be at least one zero. However, $g(\lambda)$ is in general only piecewise differentiable, so that there may be points where the derivative is discontinuous. In this case, there is a possibility that zeros do not exist. In fact, zeros always will exist, and we have the following theorem.

Theorem 3a: Let $\varphi(\lambda)\equiv \frac{d\mu}{d\lambda}-\frac{dg(\lambda)}{d\lambda}$, where the derivative is a left-hand derivative. If $\varphi(\lambda_1)>0$ and $\varphi(\lambda_2)<0$, then $\varphi(\lambda)$ must have at least one zero in the interval $[\lambda_1,\lambda_2]$.

Proof: If $g(\lambda)$ is continuously differentiable on $[\lambda_1,\lambda_2]$ then there must be a zero. If $g(\lambda)$ is not continuously differentiable, assume that no zero exists. Then there are one or more points where $g(\lambda)$ has a discontinuity in its derivative, and a sign change must occur at one or more of these points. If this were not so, then there would have to be a sign change at a point where $\varphi(\lambda)$ is continuous, and hence there would be a zero. Call the first of these points λ_d. This being the first (smallest) value of λ where a sign change occurs, the change must be from positive to negative. Then at λ_d two of the (continuously differentiable) constraint curves which generate $g(\lambda)$ are equal. Call the two active constraints g_1 and g_2, g_1 being active for $\lambda<\lambda_d$ and g_2 being active for $\lambda>\lambda_d$. Then, since $\lim_{\lambda\to\lambda_d^-}\varphi(\lambda) > 0 > \lim_{\lambda\to\lambda_d^+}\varphi(\lambda)$, we must have

$$-\left.\frac{dg_1}{d\lambda}\right|_{\lambda=\lambda_d} > -\left.\frac{dg_2}{d\lambda}\right|_{\lambda=\lambda_d} \quad\text{or}\quad \left.\frac{dg_1}{d\lambda}\right|_{\lambda=\lambda_d} < \left.\frac{dg_2}{d\lambda}\right|_{\lambda=\lambda_d}$$

But then, for some $\varepsilon>0$, we would have

$$g_1(\lambda_d)+\varepsilon g_1'(\lambda_d) < g_1(\lambda_d)+\varepsilon g_2'(\lambda_d) = g_2(\lambda_d)+\varepsilon g_2'(\lambda_d)$$

Except for higher order terms, this is just $g_1(\lambda_d+\varepsilon) < g_2(\lambda_d+\varepsilon)$. Since $g(\lambda)=\min g_i(\lambda)$, this means that g_1 is the active constraint for some $\lambda>\lambda_d^+$, contrary to hypothesis. Therefore there must be at least one zero of $\varphi(\lambda)$. The graphical meaning of this theorem is illustrated in Figure 7. Note from the figure that any cusps in $g(\lambda)$ must point *out* of the admissible region. Also note that $\varphi(\lambda)$ is piecewise continuous, and that at all discontinuities of $\varphi(\lambda)$, the curve jumps upward. Q.E.D.

In order to prove that ACOTNF terminates, we must make some assumptions about the form of the functions $f^i(\lambda)$. 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. Informally, since the inertia matrix, Coriolis array, gravitational force vector, etc. are all analytic functions, and since (by assumption) the path constraint functions are piecewise analytic, all of the functions we have dealt with are also piecewise analytic. The function $\varphi(\lambda)$, being piecewise analytic, will therefore either be identically zero in each piece or will have a finite number of zeros in each piece. Because the trajectory will follow the boundary if $\varphi(\lambda)$ is identically zero over some interval, stopping at the end of the interval, the identically-zero intervals cause no problems. Only the rightmost point of the interval could possibly be a switching point, so having a finite number of such intervals only causes a finite number of iterations of ACOTNF. Thus convergence is assured, since there are a finite number of analytic pieces. More formally, we have the following theorem:

Theorem 3b: If the functions f^i are composed of a finite number of analytic, real-valued pieces, then the function $\varphi(\lambda)$ 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 $\mathbf{q}^i$, and since the $f^i(\lambda)$ are analytic in λ, the inertia matrix, etc. when expressed as as functions of λ (as in Eqs. (4a) and (4b)) are piecewise analytic and have a finite number of analytic pieces. The functions M,Q,R,S of Eq. (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 $J_{ij}\frac{df^i}{d\lambda}\frac{df^j}{d\lambda} = M_i\frac{df^i}{d\lambda} = 0$ which is not allowed by hypothesis. If only one of the M_i is non-zero, 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 Eq. (17b) (with "$\geq$" replaced by "$=$") for some pair of indices i and j. Since the coefficients A,B,C, and D in Eq. (17b) are analytic except at the zeros of the M_i, and because 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 (17b) holds. Since Eq. (17b) determines μ as an analytic function of λ within this interval, the bounding curve $g(\lambda)$ is piecewise analytic. The curve $\varphi(\lambda)$, then, is also piecewise analytic and is either identically zero or has a finite number of zeros in each subinterval. Thus, since $\varphi(\lambda)$ 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.

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

Proof: Proof of this theorem is straightforward. Consider a trajectory which has a smaller travel time than the trajectory produced by the ACOTNF algorithm. From equation (8), there must be some λ such that the point (λ,μ') on the new trajectory is higher than the point (λ,μ) on the ACOTNF trajectory, i.e. $\mu'>\mu$. Otherwise, we would not have a trajectory with a smaller travel time. We also know from the analysis using the maximum principle that the solution must be bang-bang, i.e. that the solution will consist of a number of curves of maximum acceleration and deceleration, so we may deal only with such bang-bang trajectories. Now there are four possibilities. (λ,μ') may lie over a the initial accelerating ACOTNF trajectory, it may lie over the final decelerating ACOTNF trajectory, or it may lie over some other accelerating or decelerating trajectory. Consider the first case. In this case, the initial value for the new trajectory must be greater than the initial value for ACOTNF. Otherwise, the new trajectory would have to accelerate faster than the ACOTNF trajectory at some point, which is impossible, since the ACOTNF trajectory has the maximum allowable acceleration. The new trajectory therefore cannot meet the proper boundary conditions. The second case is similar. Since (λ,μ') lies above the ACOTNF trajectory, it would have to decelerate faster than the ACOTNF trajectory in order to meet the same final boundary conditions. This is impossible, since ACOTNF uses the maximum deceleration. In the third case, (λ,μ') lies above some other accelerating trajectory. In this case, the trajectory which leads to (λ,μ') must emanate from the boundary of the admissible region. Otherwise, the trajectory would have to pass through the accelerating ACOTNF trajectory, since that trajectory passes through a point on the boundary. The acceleration at the point of intersection would be greater for the new trajectory than for the old one, which is, again, impossible. A similar argument holds for the last case. Either an accelerating or a decelerating trajectory starting from (λ,μ') would have to either hit the boundary of the admissible region or decelerate faster than the ACOTNF decelerating trajectory, and hence would not lead to a solution. Q.E.D.

This method of generating the optimal trajectories actually works in any case where there are no islands of inadmissibility in the phase plane, not just the zero-friction case. The whole idea is to 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 is sufficient to cause islands in the phase plane. In this case, the algorithm must be 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 ACOTNF. 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 path possible, and backtracking when necessary. More formally, the *Algorithm for Construction of Optimal Trajectories (ACOT)* is:

S1. Construct the initial accelerating trajectory. (Same as ACOTNF.)

S2. Construct the final decelerating trajectory. (Same as ACOTNF.)

S3. Calculate the function $\varphi(\lambda)$ for the edge of the admissible region and for the edges of all the islands. At each of the zeros, construct a trajectory for which the zero is a switching point, as in ACOTNF steps S5 and S6. 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}.

S4. Find all intersections of the trajectories. These are potential switching points.

S5. Starting at $\lambda=0$, $\mu=\mu_0$, traverse the grid formed by the various trajectories in such a way that the highest path 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 S3 and S4 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 lead 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:

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 ACOTNF, and will not be repeated here. Note that in the convergence proof for ACOTNF 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.

6. 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 high-friction and low-friction cases. 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.

Under the assumption that the torques or forces at the inputs of the robot are bounded we derived an algorithm for determining open-loop minimum-time controls for motion of a robot along a given spatial path. However, it should be possible to obtain solutions for different input constraints. 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 path planning algorithms can be measured.

REFERENCES

[1] D. E. Whitney, "Resolved motion rate control for manipulators and human prostheses", *IEEE Trans. on Man-Machine Systems*, vol. MMS-10, pp. 47-53, June 1969

[2] J. Y. S Luh, M. W. Walker, and R. P. C. Paul, "Resolved-acceleration control of mechanical manipulators", *IEEE Trans. on Automatic Control*, vol. AC-25, no. 3, pp. 468-474, June 1980

[3] S. Dubowsky and D. T. DesForges, "The application of model-referenced adaptive control to robot manipulators", *ASME J. DSMC*, vol. 101, pp. 193-200, September 1979.

[4] A. J. Koivo, and T. -H. Guo, "Adaptive linear controller for robotic manipulators", *IEEE Trans. on Automatic Control*, vol. AC-28, no. 2, pp. 162-170, February 1983

[5] B. K. Kim and K. G. Shin, "An adaptive model following control of robotic manipulators", to appear in *IEEE Trans. Aerospace and Electronic Systems*.

[6] J. Y. S. Luh and M. W. Walker, "Minimum-time along the path for a mechanical manipulator", *Proc. of the IEEE CDC*, Dec. 7-9, 1977, New Orleans, pp. 755-759

[7] J. Y. S. Luh, and C. S. Lin, "Optimum path planning for mechanical manipulators", *ASME Journal of Dynamic Systems, Mearurement, and Control*, vol. 2, pp. 330-335, June 1981

[8] M. E. Kahn and B. E. Roth, "The near minimum-time control of open-loop articulated kinematic chains", *ASME J. DSMC*, vol. 93, no. 3, pp. 164-172, September 1971.

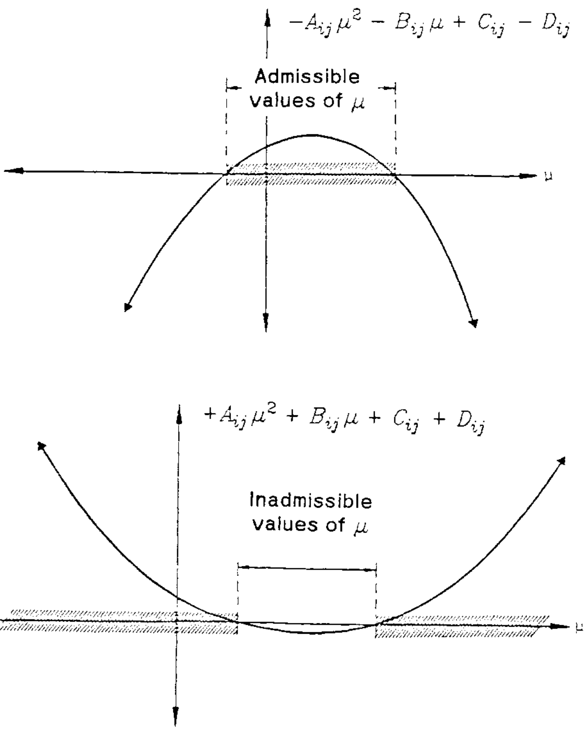
[9] B. K. Kim and K. G. Shin, "Near-optimal control of industrial manipulators with a weighted minimum time fuel criterion", to appear in *Proc. 22nd CDC*, San Antonio, TX., Dec. 1983.

[10] C.-S. Lin, P.-R. Chang, and J. Y. S. Luh, "Formulation and optimization of cubic polynomial joint trajectories for mechanical manipulators", *Proc. 21 CDC*, Orlando, FL., Dec. 1982.

[11] D. E. Kirk, *Optimal control theory: an introduction*, Prentice-Hall, Englewood Cliffs, New Jersey, 1971, pp. 227-238

[12] R. P. C. Paul, *Robot manipulators: Mathematics, programming, and control*, MIT Press, Cambridge, Mass., 1981, pp. 157-195.

[13] D. Ter Haar, *Elements of Hamiltonian mechanics*, Second edition, Pergamon Press, 1971, pp. 35-49.



$-A_{ij}\mu^2 - B_{ij}\mu + C_{ij} - D_{ij}$
Admissible values of μ
μ
$+A_{ij}\mu^2 + B_{ij}\mu + C_{ij} + D_{ij}$
Inadmissible values of μ
μ

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

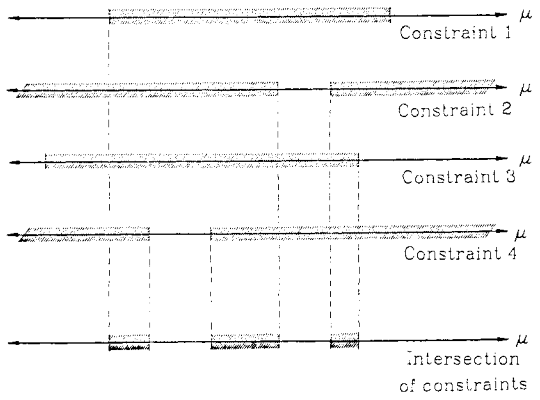
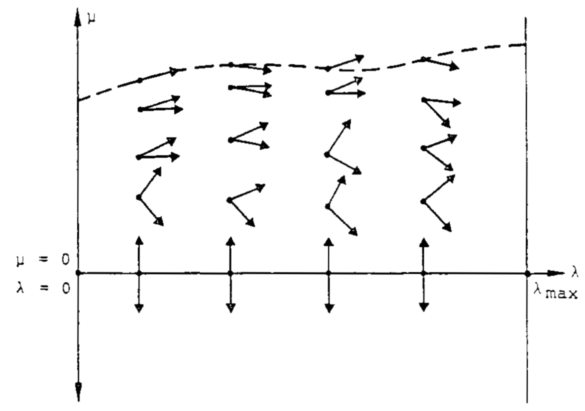
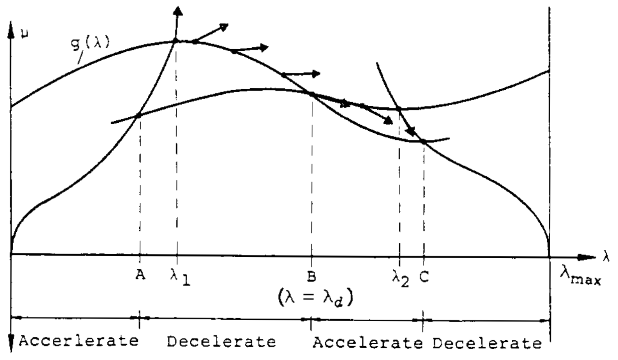


Figure 2. Intersection of admissible regions of μ



Upper arrows give maximum acceleration
Lower arrows give maximum deceleration
Dashed curve is boundary of admissible region

Figure 3. Phase portrait showing acceleration and deceleration vectors at each state



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

Figure 6. A complete optimal trajectory formed by ACOTNF with three switching points

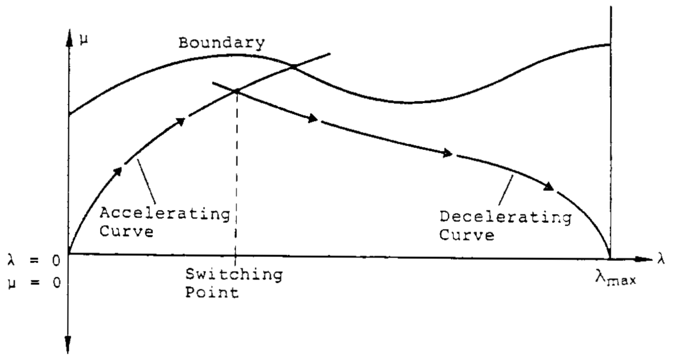
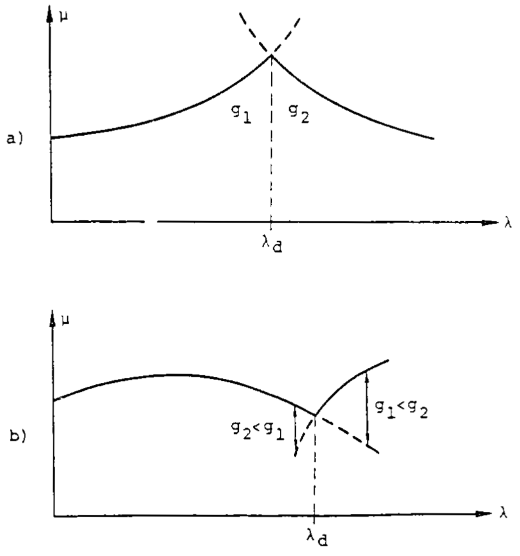
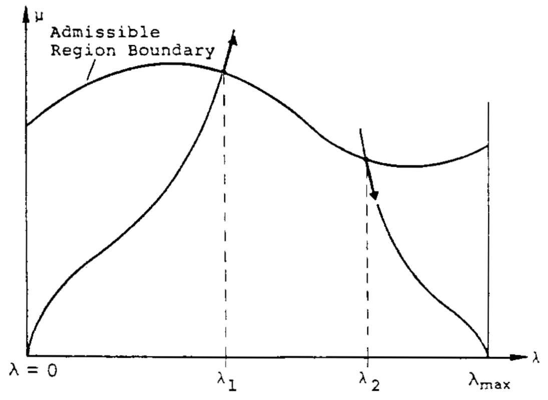


Figure 4. Case when accelerating and decelerating curves intersect (with $\mu_0 = 0$)





Situation (b) cannot occur, since we would have $g(\lambda) = g_2(\lambda) = \min(g_1(\lambda), g_2(\lambda))$ for $\lambda<\lambda_d$, contrary to hypothesis

Figure 7. Slope discontinuities in boundary curves

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