This equation leads to the following for H_2 in terms of known quantities:

$$H_2 = P - h_1 u_a - h_2 s_6 u_n - h_2 c_6 u_t. \tag{15}$$

Because (H_{2x}, H_{2y}, H_{2z}) are known from (15), (13) can be used to solve for θ_1 and θ_2. Two values of θ_2 are obtained from (13c).

$$\theta_2 = \arcsin(-H_{2z}/q_3), \quad \text{and } (\pi - \theta_2). \tag{16}$$

Then θ_1 can be found unambiguously from (13a) and (13b).

$$\theta_1 = 2 \arctan\left(\frac{H_{2y}}{H_{2x} + q_3 c_2}\right). \tag{17}$$

To determine θ_4, we consider point H_1 (Fig. 1). In this case, there is no need to proceed formally as before because the current location of point H_1 is known simply as follows:

$$H_1 = P - h_1 u_a. \tag{18}$$

Also, by applying the zero reference position method, and using $H_{10} = (0, 0, h_2)^t$, we obtain

$$\begin{bmatrix} H_1 \\ 1 \end{bmatrix} = D_1 D_2 D_3 D_4 \begin{bmatrix} H_{10} \\ 1 \end{bmatrix}. \tag{19}$$

Rearranging

$$(D_1 D_2 D_3)^{-1} \begin{bmatrix} H_1 \\ 1 \end{bmatrix} = \begin{bmatrix} 0 \\ -h_2 s_4 \\ h_2 c_4 \\ 1 \end{bmatrix}. \tag{20}$$

From (20), we find s_4 and c_4 and determine θ_4 unambiguously as

$$\theta_4 = 2 \arctan\left(\frac{H_{1x} s_1 - H_{1y} c_1}{h_2 + H_{1x} c_1 s_2 + H_{1y} s_1 s_2 + H_{1z} c_2}\right). \tag{21}$$

There are four sets of practical solutions (i.e., $q_3 > 0$). Singularities occur when $\theta_5 = 0$ or 180° (10), and $\theta_2 = \pm 90°$ (16). Investigation of (13) and (20) shows that the following relationship exists among the solution sets:

$$\begin{array}{cccccc} \theta_1 & \theta_2 & q_3 & \theta_4 & \theta_5 & \theta_6 \\ \pi + \theta_1 & \pi - \theta_2 & q_3 & \pi + \theta_4 & \theta_5 & \theta_6 \\ \theta_1' & \theta_2' & q_3 & \theta_4' & -\theta_5 & \theta_6' \\ \pi + \theta_1' & \pi - \theta_2' & q_3 & \pi + \theta_4' & -\theta_5 & \theta_6' \end{array}. \tag{22}$$

References

[1] J. Duffy, *Analysis of Mechanisms and Robot Manipulators.* New York, NY: Wiley, 1980.

[2] K. C. Gupta, "A note on position analysis of manipulators," in *Proc. 7th Applied Mechanisms Conf.* (Kansas City, MO), pp. 3.1-3.3, 1981; also see: *1984 Mechanism and Machine Theory,* vol. 19, pp. 5-8.

[3] K. C. Gupta, "Kinematic analysis of manipulators using the zero reference position description," *Int. J. Robotics Res.,* vol. 5, no. 2, pp. 5-13, 1986.

[4] K. C. Gupta and K. Kazerounian, "Improved numerical solutions of inverse kinematics of robots," in *IEEE Int. Conf. on Robotics and Automation,* pp. 743-748, 1985.

[5] F. L. Litvin and V. Parenti-Castelli, "Robot's manipulators: Simulation and identification of configurations, execution of prescribed trajectories," in *IEEE Int. Conf. Robotics,* pp. 34-44, 1984.

[6] R. P. Paul, *Robot Manipulators.* Cambridge, MA: MIT Press, 1981.

[7] D. L. Pieper, "The kinematics of manipulators under computer control," Ph.D. dissertation, Stanford University, Stanford, CA, 1968.

Optimal Robot Path Planning Using the Minimum-Time Criterion

JAMES E. BOBROW

Abstract—**A path planning technique is presented which produces time-optimal manipulator motions in a workspace containing obstacles. The full nonlinear equations of motion are used in conjunction with the actuator limitations to produce optimal trajectories. The Cartesian path of the manipulator is represented with *B*-spline polynomials, and the shape of this path is varied in a manner that minimizes the traversal time. Obstacle avoidance constraints are included in the problem through the use of distance functions. In addition to computing the optimal path, the time-optimal open-loop joint forces and corresponding joint displacements are obtained as functions of time. The examples presented show a reduction in the time required for typical motions.**

Introduction

The automatic generation of large motions for robots is a problem which must be examined carefully if truly automated factories are to become a reality. Many researchers have separated the problem into several smaller, more tractable subproblems. Some of these are: collision-free path planning, time-optimal control along specified paths, feedback control along a specified path using a known velocity profile, and vision-based path planning. In this communication, the problem of finding the collision-free path which gives the minimum-time motion is investigated. It is assumed that an initial collision-free path is known as a starting point, that the manipulator equations of motion are known, and that a geometric description of the robot workspace is available.

Research on determining the shape of the path for collision-free motions has been conducted by Luh [12], Lozano-Pérez [11], and Brooks [3]. Since infinitely many collision-free paths are usually possible, the criterion used to select one of them is most often minimum-distance, that is, the path with the smallest arc-length. In many cases this solution is not practical, since minimum-distance paths are straight lines with sharp corners near the obstacle intrusions. Robot motion along such paths would require the velocity of the arm tip to stop or slow down considerably at each corner. This leads to the need for some criterion other than minimum distance to be used for selection of the collision-free path. In this study minimum time rather than minimum distance was used as the basis for determining the path.

Because of the importance of the time-optimal control problem, several approaches have been developed for its solution in recent years. The research can be separated roughly into three categories, which are 1) determination of the minimum time velocity profile for a specified geometric path [2], [15], 2) direct use of the Maximum Principle [10] to determine the time-optimal control inputs required for unconstrained motion between two endpoints [9], [13], [19], and 3) the use of approximations of an initial feasible trajectory in conjunction with an iterative, nonlinear parameter optimization algorithm to determine either unconstrained or collision-free motions between two endpoints [4], [5], [7], [8], [14]. In addition to these three categories, research has been conducted on the selection of good geometric paths using analytical methods. In [16], it has been shown that a path in the form of a geodesic minimizes a lower bound on the traversal time.

The research reported here falls into category 3, and is similar to

Manuscript received May 12, 1987; revised November 30, 1987. This research was supported by the National Science Foundation under Grant DMC 8504928.

The author is with the Department of Mechanical Engineering, University of California, Irvine, Irvine, CA 92717.

IEEE Log Number 8820166.

that of Rajan [14] and of Dubowsky and Shiller [4]. For these approaches, the geometric path of the manipulator is represented as a set of parameterized interpolation functions, and the parameters are varied in such a way that the path traversal time required for the motion is minimized. For any specified path, the time-optimal velocity profile is found using the algorithm developed in [2], which ensures that the actuators are used to their fullest extent during the motion along the path. As an alternative to parameterizing the motion of each joint first in space and then in time, Gilbert and Johnson [7], [8] approach the same problem by parameterizing the motion of each joint directly as a function of time. The joint motions are then optimized while ensuring that at each instant, obstacles are avoided and no actuator input needed for the motion exceeds its bound.

The following sections describe in more detail the optimization approach used. A discussion is given regarding some of the numerical difficulties which were encountered, and the steps taken to avoid them. In addition, a comparison is made between the approach used here, and the time-optimal control formulation of the obstacle-free motion problem.

Problem Definition and Optimization Procedure

For the following development, the equations of motion for a three-degree-of-freedom, elbow-type robot were used. The manipulator mass properties and link lengths used for the examples in this communication are given in Appendix I. However, the procedure can be used for any nonredundant robot configuration, or for redundant-type robots if all the joint displacements have been given as specified functions of a path parameterization. The three-dimensional Cartesian path of the end-effector was represented with uniform cubic B-spline polynomials [1] (see Appendix II). Polynomials of this form have efficient computational properties, and are C^2 continuous. The parameterization for motion in the x spatial dimension is given by

$$x(s)=\sum_{i=-1}^{n+1} v_i b_i(s), \qquad 0\le s\le n \tag{1}$$

where v_i are specified path vertices, and $b_i(s)$ are piecewise-cubic B-spline basis functions. For any s, only four of the basis functions $b_i(s)$ are nonzero. This provides computational simplicity since for any value of s the summation in (1) reduces to only four terms. It also provides for local control of the path shape since any position is influenced only by its closest four neighboring vertices.

For any given initial and final path positions, the vertex values v_{-1} and v_{n+1} can be computed explicitly to ensure that the given initial position $x(0) = x_i = v_0$, and that the given final position $x(n) = x_f = v_n$. Once the remaining vertices v_i, $i = 1, 2, \cdots, n-1$ have been specified, and the remaining Cartesian coordinates $y(s)$, $z(s)$ are similarly defined, the path of the robot end-effector is defined entirely by the parameter s. From this path definition, the time-optimal velocity profile for s can be found using the algorithm presented in [2]. This algorithm makes use of the fact that for any position on the path s, a velocity $\dot{s}_{\max}$ can be found above which no combination of admissible joint torques can keep the manipulator traveling on the path. The optimum velocity profile $\dot{s}(t)$ is then found by integrating the equations of motion using the maximum or minimum available joint torques which will keep the robot traveling along the path, and which will keep the velocity below the maximum $\dot{s}_{\max}$ at every point s on the curve. The minimum-time algorithm of [2] is computationally efficient, and only requires iteration on one variable at certain positions on the phase plane.

The optimal path planning problem can then be written as one of minimizing

$$J(\boldsymbol{v})=t_f=\int_0^n \frac{ds}{\dot{s}} \text{ (the total path traversal time)} \tag{2}$$

while ensuring that the robot equations of motion

$$M(\boldsymbol{q})\ddot{\boldsymbol{q}}+\boldsymbol{b}(\boldsymbol{q},\dot{\boldsymbol{q}})=\boldsymbol{u} \tag{3}$$

are satisfied, where the individual joint torques u_i are bounded by

$$u_{i\min}(\boldsymbol{q},\dot{\boldsymbol{q}})\le u_i\le u_{i\max}(\boldsymbol{q},\dot{\boldsymbol{q}}), \qquad i=1,2,3. \tag{4}$$

The bounds $u_{i\min}$ and $u_{i\max}$ are assumed to be known functions of the joint positions and velocities, and are often constants. In addition to constraints on the joint torques, the initial and final joint positions

$$\boldsymbol{q}(t_0)=\boldsymbol{q}_0 \quad \text{and} \quad \boldsymbol{q}(t_f)=\boldsymbol{q}_f \tag{5}$$

must be reached, and it is assumed that $\dot{\boldsymbol{q}}(t_0) = \dot{\boldsymbol{q}}(t_f) = 0$. During the motion, the joint displacements are also usually bounded by

$$q_{i\min}\le q_i(t)\le q_{i\max}, \qquad i=1,2,3 \tag{6}$$

with the limits on q_i depending on the physical construction of the manipulator.

The final constraint is that of collision avoidance. This condition is given by

$$0<\delta\le d(\boldsymbol{v}) \tag{7}$$

where $d(\boldsymbol{v})$ is the minimum distance between the robot and all obstacles taken over the entire path, $\boldsymbol{v}$ is the set of $3(n+3)$ vertices which define the path given by (1), and δ is a constant which represents the margin of safety for the motion. Note that the path vertices $\boldsymbol{v}$ completely determine the end-effector path and joint positions from (1). Therefore, the distance function depends only on $\boldsymbol{v}$. For a manipulator of arbitrary shape and general obstacles, the evaluation of $d(\boldsymbol{v})$ is computationally intensive. Some progress and results on the evaluation of $d(\boldsymbol{v})$ and its gradient have been reported in [7], but a considerable amount of work remains to be completed if typical CAD solid modeling systems are to be used for the manipulator geometric representation.

The numerical optimization problem can then be stated as: Find the B-spline vertices v_i, $i = 1, 2, \cdots, n-1$ which, along with the specified values of v_0, v_n and the computed values of v_{-1} and v_{n+1}, define a path that minimizes (2) subject to (3)–(7). For any set of vertices v_i, we use the minimum-time algorithm of [2] to evaluate the traveling time t_f. This algorithm ensures that the constraints (3)–(5) are satisfied, and requires the control $u_i(t)$, $i = 1, 2, 3$, to be on the boundary of the constraint set. That is, at any instant, at least one of the control inputs $u_i(t)$ is at its maximum or minimum value. In addition to satisfying these constraints, the minimum-time algorithm of [2] produces the optimal open-loop joint angles $\boldsymbol{q}(t)$ and corresponding joint torques $\boldsymbol{u}(t)$ which will keep the robot tip moving on the three-dimensional path specified by a given set of vertices v_i.

The joint displacement constraints (6) and minimum distance constraint (7) are the only remaining constraints which must be enforced. Hence, the task of minimizing J reduces to that of varying the path vertices v_i, computing the constraints (6) and (7), and evaluating t_f using the minimum-time algorithm. Several nonlinear optimization techniques can be used for the minimization. For the examples reported in this study, the general-purpose optimization program ADS [17] was used. This program offers several different options for the optimization algorithm to be used. The algorithm which performed most satisfactorily for the examples in the communication was a sequence of unconstrained minimizations with cubic interior penalty functions for the constraints. For each unconstrained minimization, the Davidon–Fletcher–Powell search technique [6] was used. The method of Golden Sections [18] was used for the one-dimensional search.

The optimization requires the gradients of the objective function t_f, and of the constraints $d(\boldsymbol{v})$ and $q(\boldsymbol{v})$. Analytic derivatives of the minimum traversal time $\partial t_f/\partial v_i$ require differentiation of the right-hand side of (2). This, in turn, requires the derivative $\partial\dot{s}/\partial v_i$, but $\dot{s}$ is found from the solution to the time-optimal control problem as defined in [2]. It is possible to obtain the derivatives required, but the derivation needed for this is extremely tedious. The partial derivatives $\partial d(\boldsymbol{v})/\partial v_i$ are easier to evaluate, and the properties of these functions are discussed thoroughly in [7]. It is clearly demonstrated in

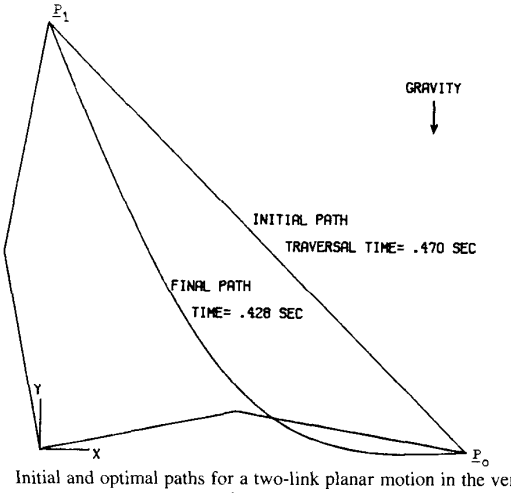


Fig. 1. Initial and optimal paths for a two-link planar motion in the vertical plane.

[7] that there may be some points (i.e., when the minimum distance is not unique) along the path where the function $d(v)$ is not differentiable. However, no problems were encountered in this study due to this lack of differentiability because these points occur only at isolated positions of the arm, and these positions were not encountered for the motions considered. The lack of differentiability $d(v)$ may present a problem for certain cases of more general paths and obstacles. The final partial derivatives needed for the optimization are $\partial q_i/\partial v_i$ and these are the easiest to evaluate since the joint angles are obtained directly from the three-dimensional path using the robot's inverse arm solution.

For the results reported here, we did not compute any of these derivatives analytically because of the difficulty in differentiating t_f. Analytic gradients would be extremely tedious to derive for the general three-degree-of-freedom robot under consideration. The ADS program allows finite-difference gradients to be computed with a user-defined step size. The time required to compute the finite difference gradients adds significantly to the overall optimization time. However, the overall computation time was relatively short. On a VAX 11-780, the CPU time for the entire optimization ranged from 2 to 30 min, depending on the number of path vertices v_i used for the path representation.

Numerical Examples

The first example considered was that of finding the optimal path between p_0 and p_1 as shown in Fig. 1. This is a planar problem which uses the full nonlinear manipulator equations of motion with the inertial properties given in Appendix I. The motion takes place in the vertical plane, so gravity is included in the model. The mass properties and link lengths used in the equations of motion were chosen to approximate UCI's experimental elbow type manipulator. The actuator torque bounds were chosen to be constant even though they are actually nonlinear functions of the joint angles. The main reason for using constant bounds is that it is easier to interpret the numerical output. Only a small amount of extra computational overhead is needed to use arbitrary torque bounds with the algorithm in [2].

In this problem we assume that there are no obstacles in the workspace, and that no joint angular displacement constraints are active. This gives rise to an *unconstrained* minimization since the inequalities (6) and (7) are both satisfied. The initial path was a straight line between p_0 and p_1. For the first optimization, the path was broken into two uniform cubic B-spline intervals with the x and y positions of the center vertex being the parameters to be varied during the minimization. After thirty iterations (including finite-difference gradients) and about 3 CPU minutes on the VAX 11-780, the curved path also shown in Fig. 1 was obtained. Fig. 2(a) and (b) shows the

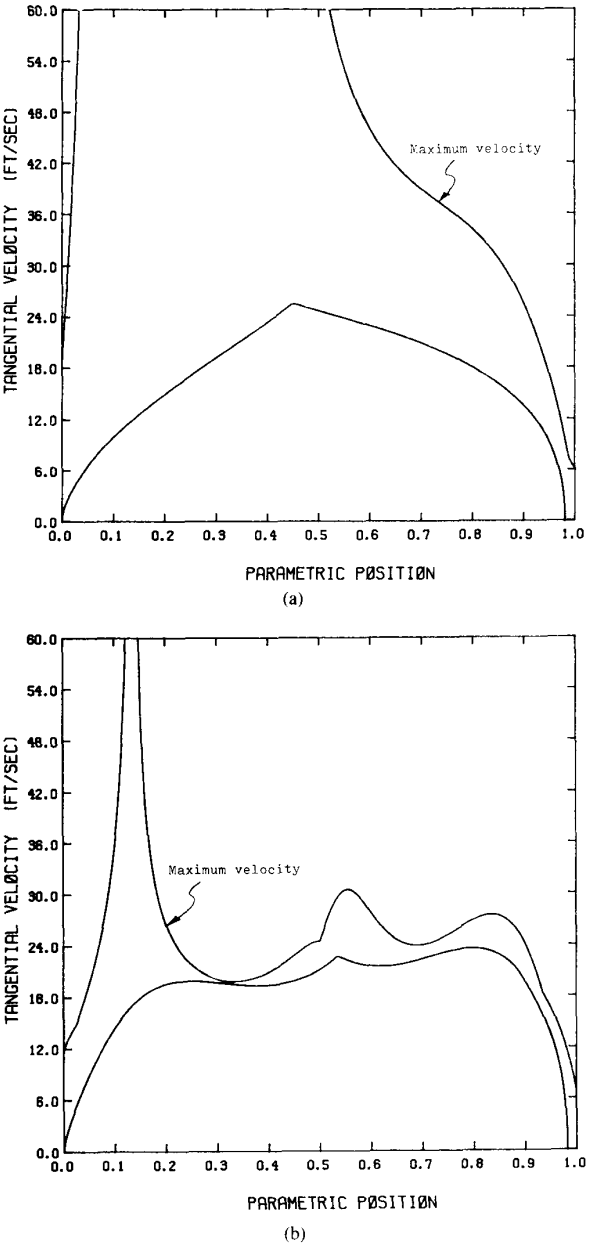


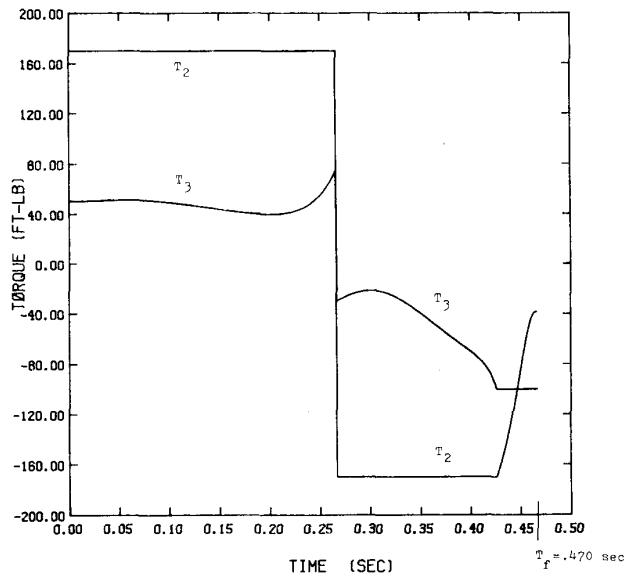
Fig. 2. (a) Optimal phase plane trajectory for the initial path of Fig. 1. (b) Optimal phase plane trajectory for the final path of Fig. 1.

optimal velocity profiles for the initial path and for the curved path. Note that the traveling time

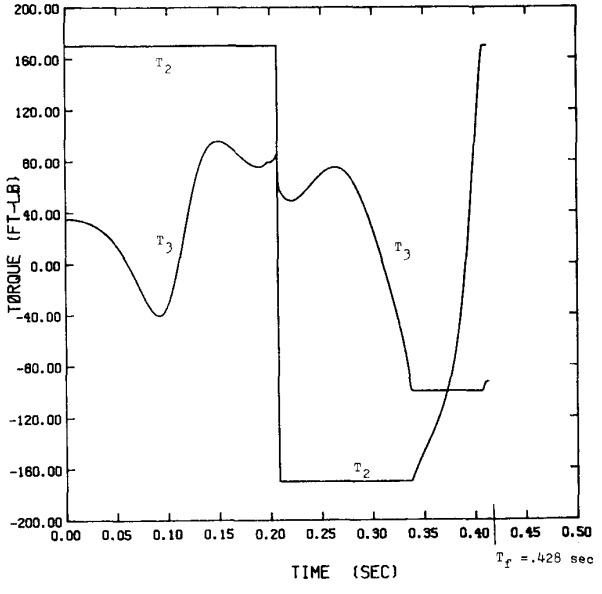
$$t_f = \int_0^n \frac{ds}{\dot{s}} = 0.47 \text{ s}$$

for the first path and $t_f = 0.428$ s for the optimal path. It can be seen that the average tangential velocity was higher for the second case.

Fig. 3(a) and (b) shows the optimal open-loop joint torques for both cases. Note that the optimal path requires a higher joint torque throughout the interval from the actuator that is not at its limit than does the initial straight-line path. This shows that the curved path produces a more uniform distribution of the actuator torques for the time-optimal trajectory. The path obtained agrees with our intuition.



(a)



(b)

Fig. 3. (a) Optimal open-loop torques corresponding to the trajectory shown in Fig. 2(a). (b) Optimal open-loop torques corresponding to the trajectory shown in Fig. 2(b).

If one were asked to lift a weight from p_0 to p_1, one would probably pull the weight in while lifting it, which is similar to the path obtained from the optimization.

In order to examine convergence properties, the path was broken into more intervals. Fig. 4 shows the optimum path when 1, 3, 5, and 8 intermediate B-spline vertices are used for the path representation. Note that the difference in the optimum traversal time between the case of 1 intermediate vertex and 8 intermediate vertices was only 2.5 percent. The computation time needed for each case increased approximately linearly with the number of vertices used. The nonlinearity of the function being minimized prohibits uniqueness arguments to be made for the path found from the minimization procedure. It does appear from Fig. 4 that the optimum path is converging as the number of path degrees of freedom are increased.

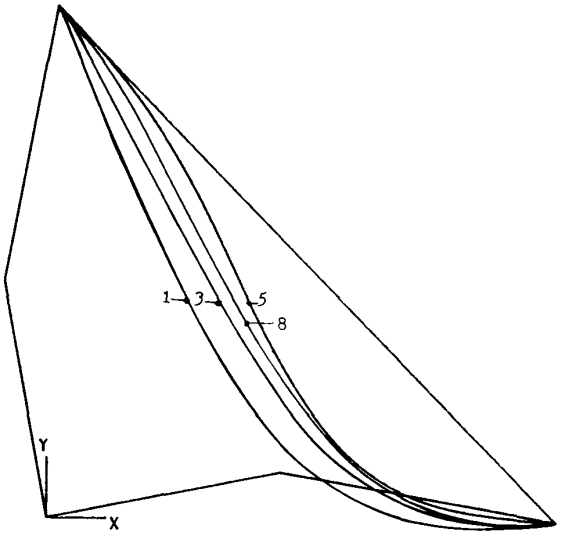


Fig. 4. The optimal paths found when 1, 3, 5, and 8 intermediate B-spline vertices were used for the path representation. The traversal times were 0.428, 0.423, 0.420, 0.418 s, respectively.

Fig. 5(a) and (b) shows the optimal velocity profile and the optimal torques obtained for the case of 8 intermediate B-spline intervals corresponding to the final path found which was shown in Fig. 4. Note that for much of the time interval, both actuator torques are nearly saturated.

The second example is that of a three-dimensional path with an obstacle in the workspace, as shown in Fig. 6. The collision avoidance constraint was enforced by ensuring that the distance between the rectangular box and the manipulator tip was always greater than zero. The distance function was evaluated by stepping the arm along the path using small increments of the path parameter s, and saving the distance $d(s)$ which was the smallest value on the interval $s \in [0, n]$. This computation was easily accomplished numerically for this simple case, but does not prevent the arm from colliding with the obstacle at points other than the tip. More sophisticated procedures must be used to obtain the distance $d(s)$ if the problem is to be solved for general shaped robots and obstacles.

Fig. 6 shows the initial path, the optimal constrained path, and the unconstrained path. Note that for the unconstrained case, the optimal path is not a straight line between the endpoints. The arm is pulled in slightly while the motion is occurring, and then it is extended. Fig. 7(a) and (b) shows the phase plane trajectories for the initial path and the final constrained paths. Observe how changing the path curvature to achieve minimum-time motion increases the maximum allowable velocity at the lowest points on the phase plane trajectory. Fig. 8 shows the open-loop joint torques corresponding to the trajectory shown in Fig. 7(b). In practical applications, high-bandwidth actuators are needed to produce the torques shown in Fig. 8 with adequate resolution. Some investigation is currently being conducted at UCI on the development of high-performance actuators for this purpose.

Discussion

As an increasing number of B-spline intervals were used, we began to experience premature termination of the optimization program. This was due to the introduction of potentially highly curved paths arising from the more closely spaced path vertices. The high local path curvature required the robot to slow down in order to stay on the path, and this resulted in local minima for the optimization. The problem was partially eliminated using a technique known as regularization. This method is used to penalize highly curved or irregularly shaped functions in order to produce smooth curves. To regularize the minimization, we add to the objective function a

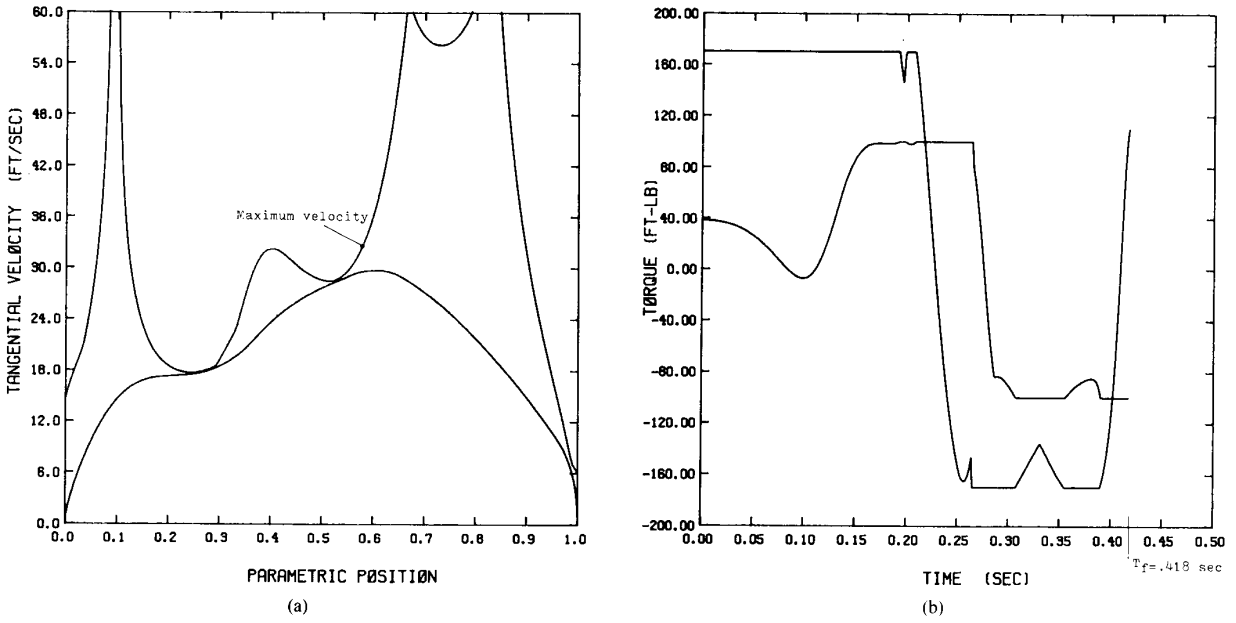


Fig. 5. (a) Optimal trajectory for the case of 8 vertices. (b) Optimal torques for the case of 8 vertices.

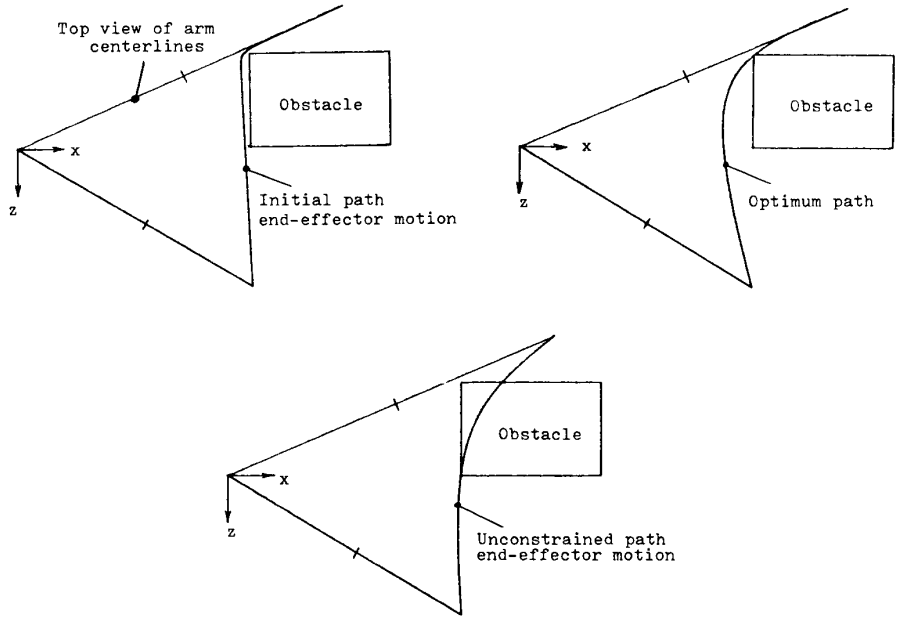


Fig. 6. Top view of three different three-dimensional motions between fixed endpoints. The traversal time for the initial path was 0.375 s, the optimal path was 0.308 s, and the unconstrained path was 0.27 s.

measure of the total path curvature

$$J(v) = t_f + \alpha C(v) \tag{8}$$

where

$$C(v) = \int_0^n \left[\left(\frac{\partial^2 x(s)}{\partial s^2} \right)^2 + \left(\frac{\partial^2 y(s)}{\partial s^2} \right)^2 + \left(\frac{\partial^2 z(s)}{\partial s^2} \right)^2 \right] ds. \tag{9}$$

The function $C(v)$ is easily shown to be quadratic in the path vertices, and can be computed analytically for any path. The scalar α is a small positive weighting factor. For the cases we have examined $\alpha = 1 \times 10^{-4}$ has worked nicely.

The question "Has the path converged to the true optimum?" is very difficult—if not impossible—to answer. As was shown in Fig. 4, the path did appear to converge as the number of intervals was increased. The optimality conditions for the unconstrained path, fixed endpoint, minimum-time motion problem can be derived using the Maximum Principle [10], and they offer some insight to the

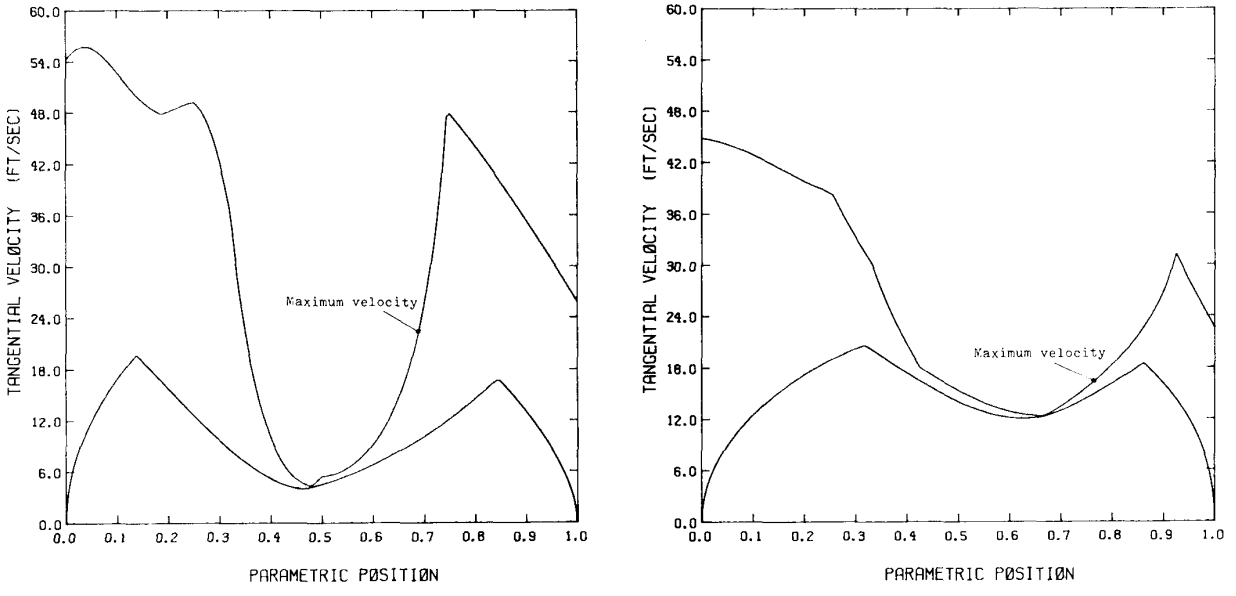


Fig. 7. (a) Time-optimal trajectory for the initial path of Fig. 6. (b) Time-optimal trajectory for the optimum path of Fig. 6.

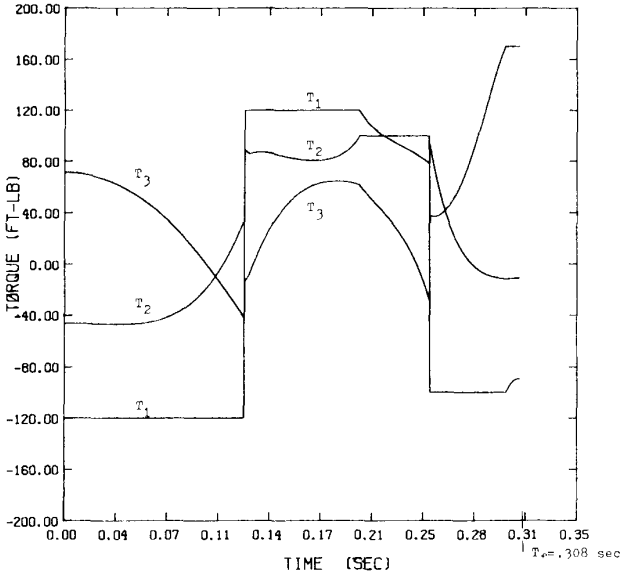


Fig. 8. Optimal open-loop torques corresponding to the trajectory in Fig. 7(b).

question's answer. To minimize the time (2), subject to the boundary conditions (5) with no constraints on the path, we obtain the standard two-point boundary value problem. Because the Hamiltonian for this system is a linear function of the controls, the Maximum Principle can be used to show that each of the control torques will be saturated (bang-bang) throughout the motion unless singular intervals exist. For this nonlinear system, it does not appear to be possible to derive meaningful analytical conditions for which singular intervals occur.

The recent paper of Meier and Bryson [13] contains several examples of optimal solutions to the two-point boundary value problem for a planar manipulator with revolute joints. The examples given in [13] rarely have any singular intervals, even for a case very close to the first example presented in this paper. It should be expected that by allowing the path vertices to vary using a numerical optimization procedure which minimizes the time as was done in this research, we would obtain the same path as that which would be found by using the Maximum Principle for the unconstrained path minimum-time problem. However, Fig. 5(b) shows that the optimal torques are not each entirely on the boundary throughout the motion. Because it is unlikely that singular intervals occur during the motion, the path converged upon by our method probably does not satisfy the optimality conditions. As more B-spline vertices are used for the path representation, one would expect to arrive at a path which does satisfy the optimality conditions of the Maximum Principle. However, the extra computation time required for this does not warrant the small decrease in the final time which would be obtained. This can be seen from noting the small changes in t_f as more vertices are used in the first example. The traversal time is first greatly reduced with only a few vertices, and the addition of more vertices only slightly decreases the time.

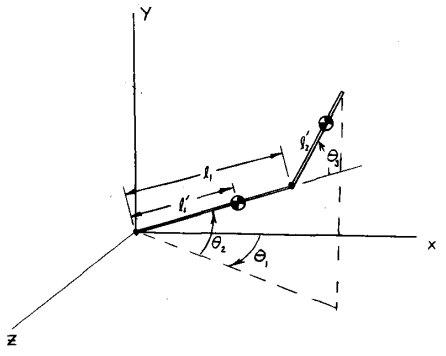


Fig. 9. Schematic representation of the manipulator model used.

Conclusions

An off-liine path planning technique has been presented which iteratively computes time-optimal paths. The full nonlinear equations of motion are used, assuming the manipulator has rigid links. A *B*-spline polynomial representation is used to represent the geometric path. The path geometry is optimized by using a general-purpose nonlinear constrained optimization program. Obstacle avoidance constraints are included in the optimization program through the use of distance functions.

Computational efficiency was not the main concern of the research reported here. However, the computation time required ranged from only 2 CPU minutes to a maximum of 30 CPU minutes on a VAX 11-780, depending on the number of vertices used to represent the path. This computation time can be significantly reduced through the use of analytic gradients of the objective function. The algorithm can be used for general off-line programming applications when the robot equations of motion are available, and a geometric description of the workspace geometry is known.

Appendix I

The manipulator used for this study was a three-degree-of-freedom elbow type as shown in Fig. 9. The mass properties and link lengths were chosen to approximate the University of California, Irvine's experimental robot. The following lists give the model parameters used for the analysis.

Link 1: Approximate Properties

Link length $l_1 = 2.021$ ft
distance to mass center $l_1' = 1.333$ ft
weight = 10.0 lb
inertias (slug·ft^2)
$I_{yy} = 0.20$
$I_{zz} = 0.20$
$I_{xx} = I_{xy} = I_{yz} = 0.0$.

Link 2: Approximate Properties

Link length $l_2 = 2.354$ ft
distance to mass center $l_2' = 1.75$ ft
weight = 13.0 lb.
inertias (slug·ft^2)
$I_{yy} = 0.10$
$I_{zz} = 0.10$
$I_{xx} = I_{xy} = I_{yz} = 0.0$.

Base Intertia

$J = 0.50$ slug·ft^2.

Torque Bounds

Base rotary actuator	$\lvert T_1\rvert \le 120$ ft·lb
lower link 1 actuator	$\lvert T_2\rvert \le 170$ ft·lb
upper link 2 actuator	$\lvert T_3\rvert \le 100$ ft·lb.

Appendix II

The uniform cubic *B*-spline basis functions used were chosen because of their simplicity and computational efficiency. An explicit formula is given for the spline function in this section. For more details on the properties of *B*-splines, see [1].

The path parameterization for the x spatial dimension is given by

$$x(s)=\sum_{i=-1}^{n+1} b_i(s)v_i, \qquad 0\le s\le n \tag{A1}$$

where the vertices v_i are the amplitudes modulating the basis functions $b_i(s)$. The desired initial end-effector position in x is v_0 and the final position is v_n. The vertices v_{-1} and v_{n+1} are given by

$$v_{-1} = 2v_0 - v_1 \tag{A2}$$

v_0 desired initial position
v_n desired final position

$$v_{n+1} = 2v_n - v_{n-1} \tag{A3}$$

where v_1 and v_{n-1} are intermediate vertices which may be any value. The purpose of v_{-1} and v_{n+1} is to cause $x(0) = v_0$ the initial value, and $x(n) = v_n$ the final value given by (A1).

The basis functions $b_i(s)$ are nonzero over four successive intervals in s, which are given by

$$b_i(s)=\begin{cases} 0, & s<i-2 \\ \frac{1}{6}(s-i+2)^3, & i-2\le s<i-1 \\ \frac{1}{6}[1+3(s-i+1)+3(s-i+1)^2-3(s-i+1)^3], & i-1\le s<i \\ \frac{1}{6}[4-6(s-i)^2+3(s-i)^3], & i\le s<i+1 \\ \frac{1}{6}[1-3(s-i-1)+3(s-i-1)^2-(s-i-1)^3], & i+1\le s<i+2 \\ 0, & i+2\le s. \end{cases} \tag{A4}$$

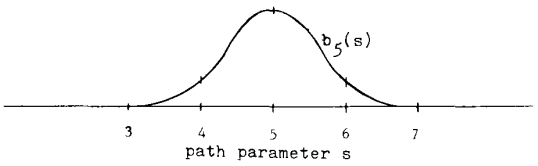


Fig. 10. The B-spline basis function $b_5(s)$.

For example, if $i = 5$, a plot of $b_5(s)$ is shown in Fig. 10. The function $b_6(s)$ is simply a copy of $b_5(s)$ shifted to the right by one interval. Note that for any s, only four of the b_i in (A1) will be nonzero. This property allows one to vary the curvature in certain portions of the path, without affecting others.

REFERENCES

[1] R. H. Bartels, J. C. Beatty and B. A. Barsky, "An introduction to the use of splines in computer graphics," U. C. Berkeley Tech. Rep. TR CSD, 83/136, Aug. 1983.

[2] J. E. Bobrow, S. Dubowsky and J. S. Gibson, "Time-optimal control of robotic manipulators along specified paths," *Int. J. Robotics Res.*, vol. 4, no. 3, Fall 1985.

[3] R. A. Brooks, "Solving the find-path problem by good representation of free space," in *Proc. AAAI 2nd Annu. Nat. Conf. on Artificial Intelligence* (Pittsburgh, PA, Aug. 18-20, 1982), pp. 381-386.

[4] S. Dubowsky and Z. Shiller, "Optimal dynamic trajectories for robotic manipulators," in *Proc. V, CISM-IFToMM Symp. on Theory and Practice of Robots and Manipulators* (Udine, Italy, 1984).

[5] S. Dubowsky, M. A. Norris and A. Shiller, "Time optimal trajectory planning for robotic manipulators," in *1986 IEEE Conf. on Robotics and Automation* (San Francisco, CA, Apr. 1986), pp. 1906-1912.

[6] R. Fletcher and M. J. D. Powell, "A rapidly convergent method for minimization," *Comput. J.*, vol. 6, no. 2, pp. 163-168, 1963.

[7] E. G. Gilbert and D. W. Johnson, "Distance functions and their application to robot path planning in the presence of obstacles," *IEEE J. Robotics Automat.*, vol. RA-1, pp. 21-30, Mar. 1985.

[8] D. W. Johnson and E. G. Gilbert, "Minimum-time robot path planning in the presence of obstacles," in *24th Conf. on Decision and Control* (Ft. Lauderdale, FL, Dec. 1985), pp. 1748-1753.

[9] M. E. Kahn and B. Roth, "The near-minimum time control of open-loop articulated kinematic chains," *ASME J. Dynamic Syst., Meas., Contr.*, vol. 93, pp. 164-172, Sept. 1971.

[10] D. E. Kirk, *Optimal Control Theory*. Englewood Cliffs, NJ: Prentice-Hall, 1970, pp. 245-246.

[11] T. Lozano-Pérez, "Automatic planning of manipulator transfer movements," *IEEE Trans. Syst. Man, Cybern.*, vol. SMC-11, pp. 681-698, Oct. 1981.

[12] J. Y. S. Luh and C. E. Campbell, "Minimum distance collision-free path planning for industrial robots with a prismatic joint," *IEEE Trans. Automat. Contr.*, vol. AC-29, pp. 675-680, Aug. 1984.

[13] E. B. Meier and A. E. Bryson, "An efficient algorithm for time-optimal control of a two-link manipulator," in *AIAA Conf. on Guidance and Control* (Monterey, CA, Aug. 1987), pp. 204-212.

[14] V. T. Rajan, "Minimum time trajectory planning," in *1985 IEEE Conf. on Robotics and Automation* (St. Louis, MO, Mar. 1985), pp. 759-764.

[15] E. G. Shin and N. D. McKay, "Minimum-time control of robotic manipulators with geometric path constraints," *IEEE Trans. Automat. Contr.*, vol. AC-30, pp. 531-541, June 1985.

[16] K. G. Shin and N. D. McKay, "Selection of near-minimum time geometric paths for robotic manipulators," *IEEE Trans. Automat. Contr.*, vol. AC-31, no. 6, pp. 501-511, June 1986.

[17] G. N. Vanderplaats, ADS, "A Fortran program for automated design synthesis, Version O.O," Naval Postgraduate School, Monterey CA, 1983.

[18] G. N. Vanderplaats, *Numerical Optimization Techniques for Engineering Design: With Applications*. New York, NY: McGraw-Hill, 1984.

[19] A. Weinreb and A. E. Bryson, "Optimal control of systems with hard control bounds," *IEEE Trans. Automat. Contr.*, vol. AC-30, no. 11, pp. 1135-1138, Nov. 1985.

On Terrain Model Acquisition by a Point Robot Amidst Polyhedral Obstacles

NAGESWARA S. V. RAO, S. S. IYENGAR, B. JOHN OOMMEN, AND R. L. KASHYAP

***Abstract*—We consider the problem of terrain model acquisition by a roving point placed in an *unknown* terrain populated by stationary polyhedral obstacles in two/three dimensions. The motivation for this problem is that after the terrain model is completely acquired, navigation from a source point to a destination point can be achieved along the collision-free paths. And this can be done without the usage of sensors by applying the existing techniques for the well-known find-path problem. In this communication, the Point Robot Autonomous Machine (PRAM) is used as a simplified abstract model for real-life roving robots. We present an algorithm that enables PRAM to autonomously acquire the model of an unexplored obstacle terrain composed of an unknown number of polyhedral obstacles in two/three dimensions. In our method, PRAM undertakes a systematic exploration of the obstacle terrain with its sensor that detects all the edges and vertices visible from the present location, and builds the complete obstacle terrain model.**

I. INTRODUCTION

In recent times there has been an enormous spurt of research activity in the algorithmic aspects of motion planning. The problem of navigating a body through a terrain populated by a set of *known* obstacles (i.e., the precise geometric characterization of the obstacles is available) is solved in many cases. Lozano-Pérez and Wesley [3], O'Dunlaing and Yap [5], Reif [7], and Schwartz and Sharir [8] present some of the most fundamental solutions to this problem. Whitesides [10] presents a comprehensive treatment on these and other solutions to the find-path and related problems. In all these methods, the precise model of the obstacle terrain is *known a priori*, and path planning is done entirely computationally. Once a path is planned, the robot moves along the planned path, and no sensors are used for navigational purposes.

Another interesting problem is the navigation of a robot in an unexplored or a partially explored terrain. In this case, the entire terrain model may not be known, and the robot relies on its sensors for navigation. Lumelsky and Stepanov [4] present sensor-based navigation algorithms for navigating a point automaton to a destination point using "touch" type of sensor. In this method localized sensor information is used to guide the point automaton, and this information is not put to any further global use. In many applications, *incidental learning* is shown to be an important enhancement in the navigation planning. Here, a composite model of the terrain is built by integrating the sensor information obtained as the robot executes sensor-based and goal-directed navigation. Iyengar *et al.* [2], Oommen *et al.* [6], Turchan and Wong [9] discuss different versions of learned navigation in unexplored terrains. Here we consider the problem of acquiring the terrain model by systematic exploration of the terrain using a sensor. Our main motivation stems from the fact that the availability of the terrain model enables us to plan the entire

Manuscript received February 26, 1987; revised December 7, 1987. A preleminary version of this paper was presented at the 3rd IEEE Conference on AI Applications, Orlando, FL, Feb. 1987. The work of B. J. Oomman was partially supported by the National Sciences and Engineering Council of Canada.

N. S. V. Rao is with the Department of Computer Science, Old Dominion University, Norfolk, VA 23529-0162.

S. S. Iyengar is with the Department of Computer Science, Louisiana State University, Baton Rouge, LA 70803.

B. J. Oommen is with the School of Computer Science, Carleton University, Ottawa K1S 5B6, Canada.

R. J. Kashyap is with the Department of Electrical Engineering, Purdue University, West Lafayette, IN 47907.

IEEE Log Number 8820163.