Robotica (1998) volume 16, pp. 89–96. Printed in the United Kingdom © 1998 Cambridge University Press

Path constrained time-optimal trajectory planning for X-Y robots via an efficient dynamic programming method

Chen-Wen V. Yen and Tsong-Zen Liu

Department of Mechanical Engineering, National Sun Yat-Sen University, Kaohsiung, Taiwan, Republic of China

(Received: February 2, 1997)

SUMMARY

By exploiting an important property of the time optimal control law structure, this paper introduces a new dynamic programming approach for X-Y robot minimum time path following problem. Compared with the conventional dynamic programming method, the proposed approach significantly reduces the computational cost. The efficiency and effectiveness of the proposed approach are demonstrated via computer simulations. In addition, via experimental studies, this paper also addresses practical difficulties in implementing the minimum-time control law.

KEYWORDS: Dynamic programming; Trajectory planning; Time-optimal control.

1. INTRODUCTION

The control problem of a manipulator can be divided into two coherent subproblems – trajectory planning and motion control. The planning of the trajectory for a robot is to convert the specifications of a given task to a desired trajectory that helps the manipulator performs the task. Motivated by the obvious relationship between task execution time and productivity, minimum-time control (MTC) problem has become one of the most extensively studied robot trajectory planning problems.

The MTC problem of robotic manipulators is concerned with the determination of control signals to drive the manipulator from a given initial condition to a desired target condition in shortest time. Based on the existence of path constraint, the MTC problem for robot systems can be partitioned into two categories: (1) path following motion;[1–11] and (2) point-to-point motion.[12–19] The focus of this work is on the problem of the first category.

Solution approaches for the path following MTC problem include path parameter optimization and dynamic programming (DP) methods. By utilizing the fact that the path following motion has only one degree of freedom (DOF), the path parameter optimization methods[1–2,4,6–11] transform the dynamic equations of the robot into the path coordinate which yields a set of equations of path parameter $s(t)$. Similarly, the joint force/torque constraints can also be transformed into constraints on the pseudo-acceleration $d^2s(t)/dt^2$. Thus, the time-optimal solution can be obtained by finding optimal pseudo-acceleration time response to produce the largest pseudo-velocity ds/dt while satisfying the joint force/torque constraints. Initially, it was thought that the optimal control is bang-bang, i.e. the pseudo-acceleration always has extreme value. Later, it was discovered that this bang-bang assumption fails at singular points where at least one joint has no effect on the end-effector acceleration along the path. Ignoring the existence of singular points may result in unbounded computation time. Hence, Shiller and Lu[11] proposed a special numerical algorithm to resolve the path singularity problem. The trade-offs of this algorithm are the increases of programming complexity and computational cost. Another drawback of the parameter optimization method is that path parameter differential equation is often stiff and thus requires special numerical integration routines to ensure accuracy.[10] This can increase computational cost conisderably.

By using parametric functions to reduce the $2n$ dimensional state space for an n DOF robot to two dimensional state space to avoid the "curse of dimensionality"[20] problem, DP has also been applied to the manipulator path following MTC problem.[3,5] Compared with the path parameter optimization method, DP does not have any path singularity or numerical stiffness problem. In addition, DP is also very easy to program. However, since DP determines the optimal solution by searching the entire solution space, it is often computationally very demanding. Additionally, as will be shown in the following section, previously developed DP methods also have difficulties in satisfying prespecified initial conditions.

The first goal of this work is to develop a new DP method for the X-Y robot path following MTC problem. Compared with the early DP methods, the proposed method has two novel features. First, it guarantees the satisfaction of initial conditions. Second, by utilizing an important property of the MTC law structure, the proposed method significantly reduces the computational requirement. The second goal of this work is to experimentally investigate the practicability of the MTC law.

This paper is organized as follows: Section 2 gives the problem statement and reviews a previously developed DP method for time optimal robot path following problem. The proposed method is introduced in Section 3. Section 4 demonstrates the effectiveness and efficiency of the proposed approach by means of computer

https://doi.org/10.1017/S0263574798000174 Published online by Cambridge University Press

simulations. Via experimental studies, Section 5 addresses practical difficulties for implementing the MTC law. Conclusions are given in Section 6.

2. PROBLEM STATEMENT AND PREVIOUS WORK

2.1 Problem statement

By adopting viscous and Coulomb friction models, the DC motor actuated X-Y robot considered in this paper can be represented by:[10]

$$\begin{aligned} \ddot{x} &= -d_x\dot{x} - c_x \operatorname{sign}(\dot{x}) + b_x u_x \\ \ddot{y} &= -d_y\dot{y} - c_y \operatorname{sign}(\dot{y}) + b_y u_y \end{aligned} \tag{1}$$

where x and y represent the displacement variables of the horizontal and vertical axes respectively. Also in equation (1), d_x and d_y are the viscous friction coefficients, and c_x and c_y are the Coulomb friction coefficients. By assuming DC motor to be armature controlled, the control variables u_x and u_y represent the armature voltages. The maximum magnitudes of these two control variables are denoted by $u_{x\max}$ and $u_{y\max}$, respectively.

With T denoting the final time of the control process and $s(t)$ representing the path parameter, the goal is to find the optimal control law to minimize the performance index

$$J = \int_0^T 1\, dt \tag{2}$$

while satisfying and the path constraints

$$x(t) = g(s(t)) \quad y(t) = h(s(t)) \tag{3}$$

and the bounded input constraints,

$$|u_x| \le r_x u_{x\max} \quad |u_y| \le r_y u_{y\max} \tag{4}$$

with $0 < r_x \le 1$ and $0 < r_y \le 1$. In addition, since the velocities at both ends of the path are typically known in practice, the path constraints of equation (3) should therefore conform to the following boundary conditions:

$$\begin{aligned} \dot{x}(0) &= \dot{x}_0 \quad \dot{x}(T) = \dot{x}_T \\ \dot{y}(0) &= \dot{y}_0 \quad \dot{y}(T) = \dot{y}_T \end{aligned} \tag{5}$$

As a final remark of the problem statement, one should note that r_x and r_y of equation (4) were chosen as unity in the previous work. However, this study relaxes this specification so that more conservative time-optimal control law can be developed. For convenience, r_x and r_y will be referred as safety coefficients in the remaining part of this paper.

2.2 Previous work

With slight differences in trajectory representation and derivation process, the ideas of the previously proposed DP methods for robot time optimal path following problem are basically the same. However, since the method developed by Singh and Leu[5] is more flexible in path representation, this work will use it as the foundation for further improvements. In particular, the remaining part of this section will expand the illustration presented by Singh and Leu[5] so that further improvements can be easily made. Specifically, after adapting to the X-Y robot systems, this algorithm consists of the following steps:

(i) Define displacement vector $\mathbf{q}$, velocity vector $\mathbf{v}$, acceleration vector $\mathbf{a}$ and control vector $\mathbf{u}$ as:

$$\begin{aligned} \mathbf{q}^T &= [q_1 \quad q_2] = [x \quad y] \\ \mathbf{v}^T &= [v_1 \quad v_2] = [\dot{x} \quad \dot{y}] \\ \mathbf{a}^T &= [a_1 \quad a_2] = [\ddot{x} \quad \ddot{y}] \\ \mathbf{u}^T &= [u_1 \quad u_2] = [u_x \quad u_y] \end{aligned} \tag{6}$$

Note that the state vector of the X-Y robot is chosen as $[\mathbf{q} \quad \mathbf{v}]^T$.

(ii) Discretize the path into $K+1$ path points. The distances between the consecutive path points should be small enough such that the acceleration in each of the path point interval can be approximated by:

$$a_i(k) \approx \frac{v_i^2(k+1) - v_i^2(k)}{q_i(k+1) - q_i(k)} \quad k = 0, \ldots, K-1 \tag{7}$$

where $q_i(k) = q_i(k\Delta t)$.

(iii) Determine $\mathbf{q}(k)$ for $k = 0, \ldots, K$.

(iv) For each DOF of the robot, specify its velocity bounds and divide the admissible velocity region into a finite number of grid points. Let $Z_i(k)$ denote the set of admissible velocity grid points associated with $q_i(k)$.

(v) Select a displacement variable q_I and designate the other displacement variable as q_J. For convenience, the axes associated with q_I and q_J will be referred as active and passive coordinates, respectively.

(vi) For every $v_I(K-1)$ grid point, use the following trapezoidal rule to calculate the time of travel between the last two path points:

$$\Delta t(v_I(K-1)) = \frac{2[q_I(K) - q_I(K-1)]}{v_I(K) + v_I(K-1)} \tag{8}$$

Note that this Δt function has only one independent variable $v_I(K-1)$ since the values of $q_I(K-1)$ and $q_I(K)$ are given by the path and the terminal velocity $v_I(K)$ is also known.

(vii) Use equations (1) and (8) to compute control $u_I(K-1)$ for every $v_I(K-1)$ grid point by approximating $a_I(K)$ with equation (7). If the resulting $u_I(K-1)$ violates the bounded input constraints of equation (4), then the corresponding $v_I(K-1)$ grid point is considered inadmissible and should be removed from $Z_I(K-1)$.

(viii) For every admissible $v_I(K-1)$ grid point, use the following trapezoidal rule to calculate $v_J(K-1)$:

$$v_J(K-1) = \frac{2[q_J(K) - q_J(K-1)]}{\Delta t(v_I(K-1))} - v_J(K) \tag{9}$$

Remove the $v_I(K-1)$ grid point from $Z_I(K-1)$ if the corresponding $v_J(K-1)$ is inadmissible. For

https://doi.org/10.1017/S0263574798000174 Published online by Cambridge University Press

the purpose of future references, record $v_J(K-1)$ as a function of $v_I(K-1)$.

(ix) Similar to step (vii), compute control $u_J(K-1)$. Remove the corresponding $v_I(K-1)$ grid point from $Z_I(K-1)$ if the bounded input constraints of equation (4) is violated.

(x) From the previous steps, it is clear that there is a unique control policy associated with every admissible $v_I(K-1)$ grid point. Hence, one can set up the following performance index table for $Z_I(K-1)$:

$$J^*_{K-1,K}(v_I(K-1)) = J_{K-1,K}(v_I(K-1)) = \Delta t(v_I(K-1)) \quad (10)$$

where $J_{K-1,K}$ is the time cost associated with the last path interval. Note that the superscript * signifies that the corresponding variable has the optimal value.

(xi) Let $k = K-2$

(xii) Use the following recurrence relation to determine the MTC law and the time cost associated with the last $K-k$ path intervals for every $v_I(k)$ grid point

$$J^*_{k,K}(v_I(k)) = \min_{v_I(k+1)\in Z_I(k+1)} [\Delta t_k + J^*_{k+1,K}(v_I(K+1))] \quad (11)$$

Note that Δt_ks are calculated from:

$$\Delta t_k = \Delta t(v_I(k), v_I(k+1)) = \frac{2[q_I(k+1) - q_I(k)]}{v_I(k+1) + v_I(k)} \quad k = 1, \ldots, K-2 \quad (12)$$

$$\Delta t_0 = \Delta t(v_I(1)) = \frac{2[q_I(1) - q_I(0)]}{v_I(1) + v_I(0)} \quad (13)$$

The reason why Δt_0 is distinguished from other Δt_ks is that Δt_0 has only one independent variable since $v_I(0)$ is known *a priori* from the prescribed path. Similar to equation (9), the corresponding $v_J(k)$ is determined from:

$$v_J(k) = \frac{2[q_J(k+1) - q_J(k)]}{\Delta t_k} - v_J(k+1) \quad (14)$$

In applying equation (11), the tested $v_I(k+1)$ should be considered inadmissible if it leads to input constraint violation or if the corresponding $v_J(k)$ exceeds one of the prespecified bounds. If no admissible $v_I(k+1)$ can be found, then the corresponding $v_I(k)$ grid point should be removed from $Z_I(k)$.

(xiii) Record the optimal solution $v^*_I(k+1)$ and $v^*_J(k+1)$ as functions of $v_I(k)$. This is necessary for subsequent applications of equation (11) and for future implementations of the MTC law.

(xiv) Let $k = k-1$ and go to step (xii) until $k = 0$.

(xv) By following the optimal control tables recorded in step (xiii) from the initial state, determine the MTC law for the control process.

An issue that has not been addressed by the above algorithm is the treatment of initial conditions. As revealed by equations (11) to (14), once the value of $v_I(k)$ is given, the value of $v_J(k)$ is also determined. In particular, when $k = 0$ we have

$$v_J(0) = \frac{2[q_J(1) - q_J(0)]}{\Delta t_0} - v_J(1) \quad (15)$$

In general, this result is not in agreement with the given initial condition of equation (5). Consequently, the MTC law determined in this manner can not satisfy the prescribed initial conditions.

3. THE PROPOSED APPROACH

3.1 Initial condition treatment

As shown in the previous section, prior DP methods attempt to transfer the X-Y robot from the given initial conditions to a path point 1 grid point in one path interval. This task can not be accomplished in the general case since it demands an arbitrary transfer of four state variables with two control commands ($u_x(0)$ and $u_y(0)$).

To resolve this problem, this work introduces an artificial path point point into the first path interval. For convenience, this artificial path point is referred as point 1/2. With δt_0 denoting the travelling time from point 0 to point 1/2 and $\delta t_{1/2}$ representing the time of travel from point 1/2 to point 1, the following formula is used to determine the MTC policy associated with the first path interval.

$$J^* = \min_{v_I(1)\in Z_I(1)} \left(\min_{\delta t_0, \delta t_{1/2}} [\delta t_0 + \delta t_{1/2} + J^*_{1,K}(v_I(1))] \right) \quad (16)$$

In implementing equation (16), the following trapezoidal rule approximations are used to compute the displacement and velocity at path point 1/2:

$$\mathbf{q}\left(\frac{1}{2}\right) = \mathbf{q}(0) + \frac{\delta t_0}{2}\left[\mathbf{v}(0) + \mathbf{v}\left(\frac{1}{2}\right)\right] \quad (17)$$

$$\mathbf{q}(1) = \mathbf{q}\left(\frac{1}{2}\right) + \frac{\delta t_{1/2}}{2}\left[\mathbf{v}\left(\frac{1}{2}\right) + \mathbf{v}(1)\right] \quad (18)$$

Solving equations (17) and (18) for $\mathbf{q}(1/2)$ and $\mathbf{v}(1/2)$ yields

$$\mathbf{q}\left(\frac{1}{2}\right) = \frac{\delta t_{1/2}\delta t_1[\mathbf{v}(0) - \mathbf{v}(1)] + \delta t_1 \mathbf{q}(0) + \delta t_{1/2}\mathbf{q}(1)}{\delta t_{1/2} + \delta t_1} \quad (19)$$

$$\mathbf{v}\left(\frac{1}{2}\right) = \frac{2[\mathbf{q}(1) - \mathbf{q}(0)] - \delta t_{1/2} v(0) - \delta t_1 v(1)}{\delta t_{1/2} + \delta t_1} \quad (20)$$

With equations (19) and (20), the controls $\mathbf{u}(0)$ and $\mathbf{u}(1/2)$ can be readily determined.

With the addition of the artificial path point, the improved DP approach can be summarized as follows:

(i) Use the following equation to set up the performance index table for $Z_I(K-1)$:

$$J^*_{K-1,K}(v_I(K-1)) = J_{K-1,K}(v_I(K-1)) = \Delta t(v_I(K-1)) = \frac{2[q_I(K) - q_I(K-1)]}{v_I(K) + v_I(K-1)} \quad (21)$$

https://doi.org/10.1017/S0263574798000174 Published online by Cambridge University Press

(ii) For all but the last and first path intervals, use the following recurrence relation to determine the MTC law for all elements of $Z_I(k)$

$$J^*_{k,K}(v_I(k)) = \min_{v_I(k+1)\in Z_I(k+1)} [\Delta t_k + J^*_{k+1,K}(v_I(k+1))]$$
$$k = 1, \ldots, K-2 \quad (22)$$

where Δt_k is computed by equation (12).

(iii) By introducing an artificial path point to divide the first path interval into two subintervals, one can determine the MTC law associated with the first path interval from

$$J^* = \min_{v_I(1)\in Z_I(1)} \left(\min_{\delta t_0, \delta t_{1/2}} [\delta t_0 + \delta t_{1/2} + J^*_{1,K}(v_I(1))] \right) \quad (23)$$

During the solution process, the velocity profile of the passive coordinate is determined by equation (14).

3.2 A new search strategy

As shown in the preceding subsection, previous DP methods determine the MTC law by searching the entire admissible active coordinate velocity region. Apparently, the efficiency of the DP methods can be improved if the size of the solution space can be reduced. Based on this concept and a property of the MTC law for robot path following problem, this subsection introduces a new search strategy to accelerate DP.

By using the Pontryagin's minimum principle to study the MTC problem for robot path following problem, Chen et al.[8] discovered that, at any instant of the control process, the MTC law requires at least one control in saturation while the remaining controls should be coordinated in such a way that the robot can stay on the desired path. Based on this property, the proposed approach replaces recurrence relation of equation (22) by:

$$J^*_{k,K}(v_I(k)) = \min_{u_x(k)=\pm r_x u_{x\max} u_y(k)=\pm r_y u_{y\max}} [\Delta t_k + J^*_{k+1,K}(v_I(k+1))] \quad k = 1, \ldots, K-2 \quad (24)$$

Instead of searching the entire admissible velocity region, this modified recurrence relation finds the optimal solution from four control saturation patterns. This simple change of search strategy, however, requires significant algorithm revision. Specifically, the modified algorithm is described as follows:

(i) Perform steps (i) to (iv) of the previous algorithm.

(ii) Perform steps (vi) to (x) of the previous algorithm with $I = 1$ and $J = 2$. Record $v_y(K-1)$ as a function of $v_x(K-1)$.

(iii) Perform steps (vi) to (x) of the previous algorithm with $I = 2$ and $J = 1$. Record $v_x(K-1)$ as a function of $v_y(K-1)$.

(iv) Combine the results of steps (iii) and (iv) to create the following performance index tables for the last path interval:

$$J^*_{K-1,K}(v_i(K-1)) = J_{K-1,K}(v_i(K-1)) = \Delta t(v_i(K-1)) \quad i = 1,2 \quad (25)$$

With these two tables, one can find the optimal value of $J^*_{K-1,K}$ via $v_x(K-1)$ or $v_y(K-1)$.

(v) Let $k = K-2$.

(vi) Set $u_x(k) = r_x u_{x\max}$, $I = 1$ and $J = 2$ then perform steps (vii) to (xii) for every $v_x(k)$ grid point.

(vii) Integrate the Ith component of equation (1) until the path point $q_I(k+1)$ is reached. If this is not possible, label the tested control pattern inadmissible. Otherwise record the resulting Δt_k.

(viii) With Δt_k obtained from the previous step, compute $v_I(k+1)$ from:

$$v_I(k+1) = \frac{2[q_I(k+1) - q_I(k)]}{\Delta t_k} - v_I(k) \quad (26)$$

The control pattern is considered inadmissible if the resulting $v_I(k+1)$ violates the velocity constraints.

(ix) With the $v_I(k+1)$ obtained from equation (26), determine the value of $v_J(k+1)$. This is feasible since $v_J(k+1)$ has been recorded as a function of $v_I(k+1)$.

(x) Compute $v_J(k)$ from:

$$v_J(k) = \frac{2[q_J(k+1) - q_J(k)]}{\Delta t_k} - v_J(k+1) \quad (27)$$

Label the tested control pattern inadmissible if the resulting $v_J(k)$ exceeds one of the velocity bounds.

(xi) With $v_J(k)$ and $v_J(k+1)$ known, compute $u_J(k)$ by approximating $a_J(k)$ with equation (7). The tested control pattern is regarded inadmissible if the resulting $u_J(k)$ violates the bounded input constraints.

(xii) If the tested control pattern is admissible, find the value for $J^*_{k+1,K}(v_I(k+1))$ from previously established performance index tables. The cost associated with tested control pattern is then computed from:

$$J_{k,k}(v_I(k)) = \Delta t_k + J^*_{k+1,K}(v_I(k+1)) \quad (28)$$

(xiii) Repeat steps (vii) to (xii) for every $v_x(k)$ grid point by using control pattern $u_x(k) = -r_x u_{x\max}$.

(xiv) Let $I = 2$ and $J = 1$.

(xv) Repeat steps (vii) to (xii) by using $u_y(k) = r_y u_{y\max}$ for every $v_y(k)$ grid point.

(xvi) Repeat steps (vii) to (xii) by using $u_y(k) = -r_y u_{y\max}$ for every $v_y(k)$ grid point.

(xvii) Based on the results obtained from steps (vii), (xiii), (xv) and (xvi), determine $v^*_x(k+1)$, $v^*_y(k)$ and $J^*_{k,K}$ for every $v_x(k)$ grid point and record them as functions $v_x(k)$.

(xviii) Based on the results obtained from steps (vi),

https://doi.org/10.1017/S0263574798000174 Published online by Cambridge University Press

(xiii), (xv) and (xvi), determine $v_y^*(k+1)$, $v_x^*(k)$ and $J_{k,K}^*$ fore every $v_y(k)$ grid point and record them as functions $v_y(k)$.

(xix) Set $k = k - 1$.

(xx) Go to step (vi) if k is not less than 1. Otherwise proceed from the next step.

(xxi) Use equation (16) to determine the MTC law associated with the first path interval and the time required to complete the control process.

(xxii) By following the optimal control tables recorded in steps (xvii) and (xviii) from initial state, determine the MTC law for the entire control process.

By comparing the two algorithms presented in this article, one finds that early DP methods perform optimal control search only on the active coordinate. In contrast, the proposed approach searches the MTC law in both coordinates. By performing searches in both axes, the time cost can be record as a function of v_x as well as a function of v_y. Therefore, performance index value can be retrieved via v_x or v_y which makes the implementation of equation (28) feasible and simple.

3.3 Computational considerations

As shown by the equations (21)–(23), the computational requirement of Singh and Leu's method is dominated by the recurrence relation of equation (22) whose computational cost is $O((K-1)p^2)$ where p is the number of grid points used to quantize the velocity region. By replacing equation (22) with equation (24), the proposed approach reduces this part of the computational cost to $O(4(K-1)p)$ since this approach only tests four control patterns.

To accelerate their algorithm, Singh and Leu proposed two techniques to improve the computational efficiency. The first technique avoids redundant search in the inadmissible velocity region by finding the bounds on the velocities from known control constraints. For the X-Y robot, the velocity bounds of the x axis can be shown to be:

$$v_{x\max} = \frac{b_x r_x u_{x\max} - c_x}{d_x} \qquad v_{x\max} = \frac{-b_x r_x u_{x\max} + c_x}{d_x} \tag{29}$$

with $v_{x\max}$ and $v_{x\min}$ denoting the upper and lower bounds of v_x. The y axis admissible velocity region can also be determined in a similar manner.

The second technique proposed by Singh and Leu is a recursive refinement procedure. Initially, this technique generates a suboptimal solution by applying a crude quantization to the admissible region. After obtaining a near optimal solution, the search region is contracted into the suboptimal solution. The process is repeated until satisfactory result is obtained. For the same accuracy, this recursive refinement procedure can reduce the computational cost of Singh and Leu's method from $O(p^{2\beta})$ to $O(\beta p^2)$ where β is number of recursions. When applied to the proposed approach, this recursive refinement procedure can reduce the computational cost from $O(p^{\beta})$ to $O(\beta p)$.

In addition to the two techniques adopted by Singh and Leu, this study uses a simple property of the time-optimal trajectory to further reduce the computational cost. In particular, it is known that the path parameter s is a nondecreasing function of time. For the X-Y robot systems, this implies that the velocity $v_i(k)$ have the same sign as $q_i(k+1) - q_i(k)$. With this property, one can reduce the size of the velocity search region by 50% and thus further improve the computational efficiency.

4. COMPUTER SIMULATIONS

The objective of this section is to compare the proposed approach with Singh and Leu's method. Note that the implementation of Singh and Leu's method is based on equations (21)–(23). The methods were tested for two paths using safety coefficients $r_x = r_y = 1$ and $r_x = r_y = 10/11$. In the first path following task, the X-Y robot is required to move in upward direction by following a straight line $x - 2y = 1$. The path length is 109.5 mm. In the second problem, the X-Y robot is required to follow a 100 mm diameter circle in counterclockwise direction.

The X-Y robot considered here consists of two identical ELITE X-Y table axes (model number EHR-D-400). The parameters of the process model were determined by a standard system identification technique. The simulations were executed on a Pentium-75 personal computer. The computer programs were written in the C language. The number of the path points and the number of velocity grid points were chosen as 50 and 1001, respectively. The recursion number of the recursive refinement procedure was selected as 5. In all simulation cases, the robot has zero velocity at the both ends of the path.

The simulation results are summarized in Table I. as shown by the simulation results, the proposed approach is as effective as Singh and Leu's method in terms of solution accuracy. However, the proposed approach is about two order of magnitudes faster than Singh and Leu's method. This clearly demonstrates the efficiency of the proposed MTC law search strategy. Additionally, it is also found that the process time has increased about 10% when safety coefficients are reduced from one to 10/11. Apparently, decreasing the value of the safety coefficients has adversely affected the efficiency of the trajectory.

6. EXPERIMENTAL STUDIES

Via experiments, the objective of this section is to investigate potential problems that may hinder the application of the MTC law. The experimental setup of the tested X-Y robot system is illustrated as follows. Two linear amplifier driven BALDOR DC motors (model number MTE3353) were used to drive the X-Y robot described in the previous section. The sampling period was chosen as 1 ms. An 80486-33 personal computer was used to handle the I/O actions. The motor velocities were measured by tachometers. Analog inputs and outputs were performed with 12-bit resolution. The repeatability of the X-Y robot axes is 10 μm. The axis

https://doi.org/10.1017/S0263574798000174 Published online by Cambridge University Press

Table I
Summary of simulation results

Simulation Cases		The Proposed Method		The Conventional Method	
Path	r_x and r_y	Computation Time (sec)	Process Time (sec)	Computation Time (sec)	Process Time (sec)
Line	1	37	1.23	5611	1.22
	10/11	37	1.36	5472	1.35
Circle	1	71	3.44	16862	3.43
	10/11	71	3.81	16495	3.80

position was measured by a linear scale which has a resolution of 5 μm.

In performing the experiments, the x and y axes were independently controlled to track the minimum-time trajectories obtained in the previous section by the proposed approach. In particular, deadbeat controllers were designed for the linear part of the system models. The nonlinear friction effect in each axis was counteracted by feedforward compensation. During the control process, the trajectory tracking error was recorded for every sampling instant. To characterize the tracking efficacy, the mean and variance of the path following error were computed. Note that this study defines the path following error as the distance between the desired and actual path points. For reliability, we average these experimental results over 5 runs. These averaged results are summarized in Table II.

As shown in Table II, safety coefficients have direct influence on the tracking accuracy. In particular, the conservative choice of $r_x = r_y = 10/11$ reduces the path following error mean by a factor of 4 in the line path case. For the circular path, this reduction factor is approximately 2. However, as shown by Table I, the selection of $r_x = r_y = 10/11$ increases the process time by about 10%. Apparently, one must balance the execution time and tracking accuracy of the trajectory by choosing appropriate values for r_x and r_y.

To gain more insight, the remaining part of this section will investigate the path error time responses for the cases of $r_x = r_y = 10/11$. In particular, Figure 1 gives the time histories of the path following error associated with the line path. As shown by Figure 1, significant tracking error occurs at both ends of the path. This phenomenon can be explained by the fact that the behavior of the DC motor is strongly nonlinear when rapid current change is in presence.[21] With the rapid current change induced by instantaneous voltage switch, the linear process model of equation (1) can not accurately characterize the DC motor dynamics at both ends of the path. This results in serious path following error at the beginning and end of the control process.

Table II
Summary of experimental results

Experimental Cases		Statistics of the Path Following Errors	
Path	r_x and r_y	Mean (mm)	Variance (mm^2)
Line	1	0.369	0.0835
	10/11	0.088	0.0168
Circle	1	0.126	0.0236
	10/11	0.060	0.0043

For the circular path, the time histories of the path error and the velocities of the X and Y axes are plotted in Figure 2. In this case, large path following error appears in the vicinity of the points where velocity of one of the axes changes direction. There are several possible explanations for this phenomenon. First, since friction opposes motion, friction switches sign as the velocity changes direction. Consequently, the one sampling period single sign friction model of equation (1) is inadequate in the sampling period where velocity changes direction. Second, it is well known that the friction effect is strongly velocity dependent in low speed.[22] Hence the deficiency of the Coulomb friction model in low speed is another possible error source. The third possible reason is related to instrumentation errors. In this study, the motor velocities were measured by tachometers which are not suitable for low speed measurements. Hence, it is reasonable to assume that the inaccurate velocity measurements also have nonnegligible impact on low speed tracking efficacy. It is important to note that, unlike path following error caused by

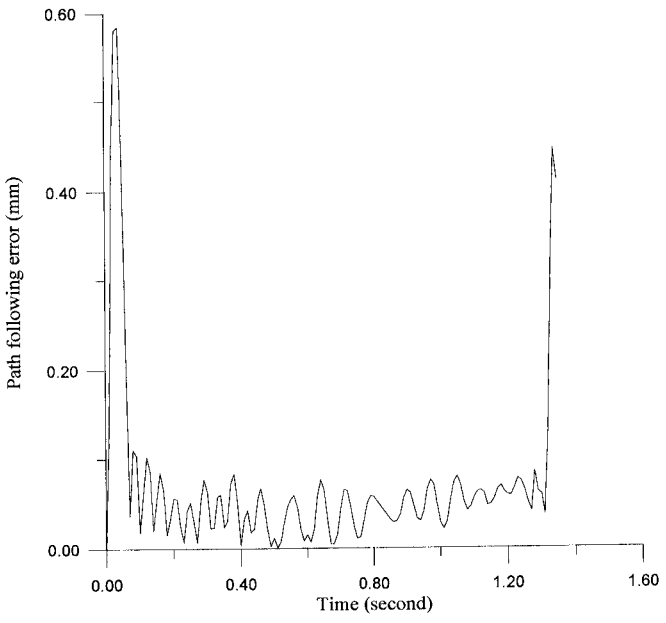


Fig. 1. Path following error of the line path.

https://doi.org/10.1017/S0263574798000174 Published online by Cambridge University Press

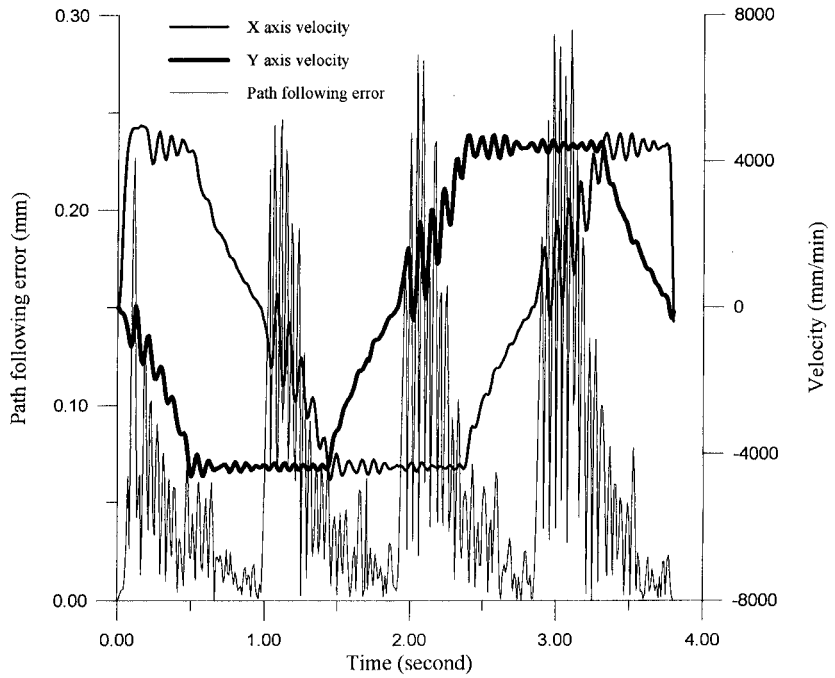


Fig. 2. Path following error and velocities of the axes.

instantaneous voltage switches, the inferior low speed tracking performance problem is not unique to time-optimal controllers. However, compounded by the fast acceleration/deceleration characteristics of the time optimal motion, the low speed tracking difficulty is particularly significant for MTC problems. In summary, nonlinearity induced by rapid control command changes and the inferior low speed tracking performance represent two important difficulties that must be surmounted before the MTC law can be accurately implemented.

7. CONCLUSIONS

Compared with the popular path parameter optimization methods, DP represents a simple and effective tool for X-Y robot time optimal path following problem since it does have any path singularity or numerical stiffness problem. By preserving these advantages, this article proposes a new DP based solution approach for X-Y robot time optimal path following problem. Compared with previous DP methods, the proposed approach has two distinct advantages. First, it guarantees the satisfaction of initial conditions. Second, the proposed approach dramatically reduces the computational cost. In our simulations, the proposed approach is about two order of magnitudes faster than a previously proposed DP method.

In addition to developing a new solution method, this work has also studied potential difficulties for time-optimal control law implementation. From experimental results, it is found that rapid control command changes occur in the time-optimal motion often result in large tracking error. Compounded by the fast acceleration/deceleration characteristics of the time-optimal motion, significant low speed path following error has also been observed. From these results, it is believed that one possible future research direction is to develop an integrated optimal trajectory planning and tracking method to implement the minimum-time control law.

ACKNOWLEDGEMENTS

This research was supported in part by National Science Council of Republic of China under Grant Number NSC 85-2212-E-110-003.

References

1. J.E. Bobrow, S. Dubrowsky and J.S. Gibson, "Time-optimal control of robotic manipulators along specified paths," *Int. J. Robotics Research* **4,** 3–17 (1985).
2. K.G. Shin and N.D. Mckay, "Minimum-time control of a robotic manipulator with actuator constraints," *IEEE Transactions on Automatic Control* **AC-30,** 531–541 (1985).
3. K.G. Shin and N.D. Mckay, "A dynamic programming approach to trajectory planning of robotic manipulators," *IEEE Transactions on Automatic Control* **AC-31,** 491–500 (1986).
4. F. Pfeiffer and R. Johanni, "A concept for manipulator trajectory planning" *IEEE Journal of Robotics and Automation* **RA-3,** 115–123 (1987).
5. S. Singh and M.C. Leu, "Optimal trajectory generation for robotic manipulators using dynamic programming," *J. Dynamic Systems, Measurement and Control* **109,** 88–96 (1987).
6. J.-J.E. Slotine and H.S. Yang, "Improving the efficiency of time-optimal path-following algorithms," *IEEE Journal of Robotics and Automation* **RA-5,** 118–124 (1989).
7. Y. Chen and A.A. Desrohers, "A proof of the structure of the minimum-time control law of robotic manipulators using a Hamiltonian formulation," *IEEE Transactions on Robotics & Automation* **6,** 388–393 (1990).
8. Y. Chen and S.Y.-P. Chien and A.A. Desrohers, "General structure of time-optimal control of robotic manipulators moving along prescribed paths," *Int. J. Control* **56,** 767–782 (1992).
9. Z. Shiller and H.H. Lu, "Computation of path constrained

https://doi.org/10.1017/S0263574798000174 Published online by Cambridge University Press

time optimal motions with dynamic singularities," *J. Dynamic systems, Measurement and Control* **114,** 34–40 (1992).
10. L.G. Van Willenburg, "Computation and implementation of digital time-optimal feedback controllers for an industrial x-y robot subjected to path, torque, and velocity constraints," *Int. J. Robotics Research* **12,** 420–433 (1993).
11. Z. Shiller, "On Singular Time-Optimal Control Along Specified Paths," *IEEE Journal of Robotics and Automation* **RA-10,** 561–566 (1994).
12. M.E. Kahn and B. Roth, "The near-minimum-time control of open-loop articulated kinematic chains," *J. Dynamic Systems, Measurement and Control* **93,** 164–172 (1971).
13. R.G. Fenton, B. Benhabib and A.A. Goldenberg, "Optimal point-to-point motion control of robots with redundant degrees of freedom," *J. Dynamic Systems, Measurement and Control* **108,** 120–126 (1986).
14. G. Sahar and J.M. Hollerbach, "Planning of minimum time trajectories for robotic arms," *Int. J. Robotic Research* **5,** 90–100 (1986).
15. K.G. Shin and N.D. Mckay, "Selection of near-minimum time geometric paths for robotic manipulators," *IEEE Transactions on Automatic Control* **AC-31,** 501–511 (1986).
16. Z. Shiller and S. Dubowsky, "Roboth path planning with obstacles, actuator, gripper and payload constraints," *Int. J. Robotic Research* **8,** 3–18 (1989).
17. Z. Shiller and S. Dubowsky, "On computing the global time-optimal motions of robotic manipulators in the presence of obstacles", *IEEE Journal of Robotics and Automation* **RA-7,** 785–797 (1991).
18. H.S. Yang and J.-J.E. Slotine, "Fast algorithms for near-minimum-time control robot manipulators," *Int. J. Robotic Research* **13,** 521–532 (1994).
19. E.A. Croft, B. Benhabib and R.G. Fenton, "Near-time optimal robot motion planning for on-line applications," *J. Robotic Systems* **12,** 553–567 (1995).
20. R.E. Bellman, *Dynamic Programming* (Princeton University Press, NJ, 1957).
21. N.K. Sinha, C.D. diCenzo and B. Szabados, "Modeling of dc motors for control applications," *IEEE Transactions on Industrial Electronics and Control Instrumentation* **IECI-21,** 84–88 (1974).
22. B. Armstrong-Hélouvry, *Control of Machines with Friction* (Kluwer Academic Press, Boston, MA, 1991).

https://doi.org/10.1017/S0263574798000174 Published online by Cambridge University Press