Proceedings of the 2001 IEEE
International Conference on Robotics & Automation
Seoul, Korea · May 21-26, 2001

A new optimization algorithm for singular and non-singular digital time-optimal control of robots

C.W.J. Hol, L.G. van Willigenburg, E.J. van Henten and G. van Straten

Systems and Control group, Wageningen University and Research Centre, Agrotechnion, Bomenweg 4, 6703 HD Wageningen, the Netherlands, E-mail : camile.hol@user.aenf.wau.nl

Abstract

Time-optimal controls for 2-link robots are often of bang-bang type. Many algorithms to solve time-optimal robot control problems a-priori assume the optimal control to be bang-bang. Industrial robots very often have 5 or 6 links and then the associated time-optimal controls are usually singular. This paper presents a new algorithm that enables computation of both bang-bang and singular time-optimal controls for robots. The algorithm uses both the conjugent gradient and Gauss-Newton method to enhance its efficiency and does not require state-parameterization, which introduces additional errors. The algorithm is used to compute time-optimal controls for an industrial 5 link robot model including gravity and viscous friction.

1 Introduction

Motion time of an articulated manipulator plays a crucial role in the productivity of robots in industrial applications. We focus in this paper on the point-to-point time-optimal control problem (PTPTOC): find the control that brings the manipulator from the initial position to the final position in minimum time, while taking into account the dynamics of the robot, constraints on the torques, friction and gravity. The robot model can be written in state-space, with the angular position and velocity as states. The PTPTOC can then be formulated as a nonlinear optimal control problem with free final time, final state constraints and bounded controls.

1.1 Related work

Many researchers have attempted to solve the PTPTOC. The developed methods can be divided into three categories: standard optimal control, control parametrization and full parametrization methods.

The standard optimal control methods use Pontryagin's Maximum Principle (PMP) to rewrite the optimal control problem into a two-point boundary value problem (TPBVP). Foutouhi-Chaouki and Szyszkowski [1] and Meier and Bryson [2] used shooting and gradient method respectively to solve the TP-BVB. These algorithms assume bang-bang control, but in the case the time-optimal control problem is singular this does not lead to optimal solutions [3]. Chen and Huang [4] computed smooth controls on singular trajectories with a sequence of non-singular problems that converge to the original problem.

Several algorithms have been developed that use control parametrization. A piecewise constant parametrization of the controls is justified by the digital nature of the controller. Van Willigenburg and Loop [3], Geering *et al.* [5] and Dissanyake *et al.* [6] present similar algorithms with this approach.

The PTPTOC for robots with >2 DOF has only been solved in the literature with the full parametrization method, which uses both state and control parametrization. Saramago and Steffen [7] use cubic polynomials to interpolate the joint angles between two sample intervals, whereas Fang and Dissanayake [8] use first order polynomials. Bezier splines are used by Dubowsky *et al.* [9] and Shiller and Dubowsky [10]. They calculate the controls that make the robot traverse the parametrized path by solving the fixed path motion planning problem. A drawback of these methods is that parametrization of the states introduces additional errors because the state trajectories do not satisfy the model equations.

Many robots in the industry have more than 2 DOF. In the case of increasing DOF the probability of time-optimal controls being singular increases, because the probability of some links having to "wait" on others increases. However, to our best knowledge all papers using standard optimal control or control parametrization present only solutions for a 2-DOF robot. Summarizing, it seems that there is no method in the literature that computes singular time-optimal controls for >2DOF robots without introducing additional errors through state parametrization. Such an algorithm is presented in this paper.

1.2 Proposed method and outline

Section 2 presents the dynamic equations of the robot model and the control problem. The optimization algorithm is described in Section 3. Starting from the gradient algorithm of Bryson [11], we added a "clipping function" to enforce the control bounds. To enhance the efficiency and accuracy a line search has been added and conjugate gradients are introduced. To further enhance the efficiency we combined the gradient method with a Gauss-Newton method. In Section 4 results are presented on a simulation model of a 5-DOF direct drive robot with viscous friction and gravity. Concluding remarks are presented in Section 5.

2 Time-optimal control problem

Consider a robotic manipulator with m rotational joints. It is assumed that the joints are rigid-body elements and are driven by direct-drive motors. If the joint angles are described by the $m \times 1$ vector θ they can be described as follows [12]:

$$M(\theta)\ddot{\theta} + V\left(\theta, \dot{\theta}\right) + G(\theta) + F\left(\dot{\theta}\right) = \tau \qquad (1)$$

where $M(\theta) \in \mathbb{R}^{m\times m}$ is symmetric positive definite inertia matrix, $V\left(\theta, \dot{\theta}\right) \in \mathbb{R}^{m\times 1}$ represents the Coriolis and centrifugal forces, $G(\theta) \in \mathbb{R}^{m\times 1}$ represents the gravity forces, $F\left(\dot{\theta}\right) \in \mathbb{R}^{m\times 1}$ are the viscous friction forces and $\tau \in \mathbb{R}^{m\times 1}$ vector of actuator torques. We consider the following model for the viscous friction in the actuator:

$$F\left(\dot{\theta}\right) = -K\dot{\theta} \qquad (2)$$

where $K \in \mathbb{R}^{m\times m}$ is a diagonal matrix with on the diagonal the positive viscous friction coefficients. The motor torques are assumed proportional to the motor current [13]. The torques are therefore considered as control variables. Bounds on the motor current are applied to prevent overheating of the motors, which can be expressed in terms of the motor torques:

$$\tau_i^{\min} \leq \tau_i \leq \tau_i^{\max} \qquad (3)$$

where $\tau_i^{\min} \in \mathbb{R}$ and $\tau_i^{\max} \in \mathbb{R}$ are the lower and upper bound for the i^{th} link respectively. By defining the joint positions and joint velocities as states ($x = \begin{bmatrix} \theta & \dot{\theta} \end{bmatrix}^T$) and the torques as controls ($\tau = u$) the following state-space representation is obtained:

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} x_2 \\ M(x_1)^{-1}\left(V(x_1, x_2) + G(x_1) + F(x_2) - u\right) \end{bmatrix} \qquad (4)$$

The number of states is twice the number of links: $n = 2m$. In the remainder of this paper the state-space system in Equation (4) is written in the short form:

$$\dot{x} = f(x, u) \qquad (5)$$

where $x \in \mathbb{R}^{n\times 1}$ and $u \in \mathbb{R}^{m\times 1}$ are the states and controls respectively and $f : \mathbb{R}^{n\times 1}\times\mathbb{R}^{m\times 1} \longrightarrow \mathbb{R}^{n\times 1}$ represents the model equations. The point-to-point motion time-optimal control problem is formulated as follows: *find the control that brings the manipulator from a specified initial position to a specified final position in minimum time, while satisfying the bounds on the controls.*

This is a nonlinear optimal control problem with free final time, fixed final state and bounded control:

Problem 1: (*time-optimal control problem: continuous-time formulation*)

$$\begin{aligned} \min_{t_{\text{final}}, u(t)} J &= \int_0^{t_{\text{final}}} 1 dt \qquad (6) \\ \text{subject to} &: \\ \dot{x} &= f(x, u) \\ x(t_{\text{initial}}) &= x_{\text{initial}}^{\text{specified}}, \quad x(t_{\text{final}}) = x_{\text{final}}^{\text{specified}} \\ \tau_i^{\min} &\leq u_i(t) \leq \tau_i^{\max} \\ i &\in \{1, 2, \ldots, m\}, \quad t \in [0, t_{\text{final}}] \end{aligned}$$

where $x_{\text{initial}}^{\text{specified}} \in \mathbb{R}^{n\times 1}$ and $x_{\text{final}}^{\text{specified}} \in \mathbb{R}^{n\times 1}$ are the specified initial and final states respectively. The decision variables of this minimization are the control trajectory $u(t)$ and the final time t_{final}.

Control parametrization Given the digital nature of the robot controller, the controls are parametrized by piecewise constants. The time is divided into N equidistant control intervals. The final time is determined by the width of the control intervals Δt: $t_{\text{final}} = N\Delta t = t_N$. The width of the control interval Δt becomes one of the degrees of freedom in the control problem. The beginning of the k^{th} interval is denoted by t_k, $k \in \{0, 1, \ldots, N-1\}$ and $t_0 = 0$ and $t_N = t_{\text{final}} = N\Delta t$. In the equations in the remainder of the paper, the time-index k relates to the time instant t_k $k \in \{0, 1, \ldots, N-1\}$ and the axis-index i should be interpreted as $i \in \{1, 2, \ldots, m\}$.

Final state We enforce the final state constraint in Problem 1 by a quadratic penalty in the objective. This is different from Weinreb and Bryson [14], who treat the final state constraint explicitly as equality constraint in their steepest descent method. They introduced extra weights on the control variables to enforce the bounds on the controls, which (as they admitted) slowed the convergence of their algorithm.

With these assumptions, Problem 1 can be cast into a discrete optimal control problem:

Problem 2: (*time-optimal control problem: discrete-time formulation*)

$$\begin{aligned}
\min_{u_0,\ldots,u_{N-1},\Delta t} J_{\text{discrete}} &= t_{\text{final}} + \left\| x_N - x_{\text{final}}^{\text{specified}} \right\|_Q \quad (7)\\
\text{subject to} &: \\
x_{k+1} &= \int_{t=t_k}^{t=t_k+\Delta t} f(x, u_k)\, dt \\
x_0 &= x_{\text{initial}}^{\text{specified}}, \quad t_{\text{final}} = N\Delta t \\
u_i^{\min} &\leq u_{ik} \leq u_i^{\max}, \quad t \in [0, t_{\text{final}}]
\end{aligned}$$

where $Q \in \mathbb{R}^{n\times n}$ is a positive diagonal weighting matrix and $\|\cdot\|_Q$ denotes the 2-norm weighted with matrix Q. A quadratic penalty is chosen to assure smoothness of the objective function close to the optimum.

3 The algorithm

The gradient algorithm ("*DOPT*") of Bryson [11] has been our starting point. Because this algorithm does not enforce bounds on the controls, we added a "clipping function". Experiments with this method showed however that this algorithm converges prohibitively slow on the time-optimal control problem of a 5-DOF robot. We introduced an advanced line search, conjugate search directions and combined the method with a Gauss-Newton method. The Gauss-Newton method exploits the specific quadratic structure of the objective function in Problem 2. First the basic algorithm will be briefly discussed. Then we will discuss our extensions.

Basic algorithm Define the Hamiltonian H as follows:

$$H(x_k, u_k, \lambda_{k+1}, \Delta t) = \lambda_{k+1}^T \int_{t=t_k}^{t=t_k+\Delta t} f(x, u_k)\, dt \quad (8)$$

where $\lambda_{k+1} \in \mathbb{R}^{n\times 1}$ is the co-state at time t_{k+1}. The algorithm of Bryson makes steps in the "steepest descent" direction. This direction is computed by differentiation of the Hamiltonian with respect to the decision variables:

$$\begin{aligned}
\delta u_k &= -\frac{\partial H\left(x_k, u_k^{\text{previous}}, \lambda_{k+1}, \Delta t^{\text{previous}}\right)}{\partial u_k} \quad (9)\\
\delta \Delta t &= -\frac{1}{N}\sum_{k=0}^{N-1} \frac{\partial H\left(x_k, u_k^{\text{previous}}, \lambda_{k+1}, \Delta t^{\text{previous}}\right)}{\partial \Delta t}
\end{aligned}$$

where u_k^{previous} and $\Delta t^{\text{previous}}$ are the decision variables at the previous iteration of the algorithm. The states are computed by integration of Equation (4) in forward direction, with the decision variables set to $u_k = u_k^{\text{previous}}$, and $\Delta t = \Delta t^{\text{previous}}$. The co-states are computed by backward recursion of the co-state equation (see Bryson [11]).

Clipping Our algorithm "clips" the controls at their extreme value to avoid that they violate the bounds. The sample time Δt is also clipped between bounds to prevent it to become negative or very large, which would cause numerical difficulties.

$$\begin{aligned}
u_k^{\text{new}} &= \min\left(u^{\max}, \max\left(u^{\min}, u_k^{\text{old}} + \alpha \delta u_k\right)\right) \quad (10)\\
\Delta t^{\text{new}} &= \min\left(\Delta t^{\max}, \max\left(\Delta t^{\min}, \Delta t_k^{\text{old}} + \alpha \delta \Delta t_k\right)\right)
\end{aligned}$$

where the functions min and max take the minimum and maximum value of each element i of two column vectors respectively. $\Delta t^{\min}$ and $\Delta t^{\max}$ are tuning parameters set at 0.005 sec and 0.15 sec respectively. $\alpha \in \mathbb{R}$ is the step-size determined with the line-search. The clipping is in accordance with Pontryagin's Maximum Principle (PMP).

Conjugate gradient It is well known that a first-order gradient method as discussed above usually shows great improvements in the first iterations but has poor convergence characteristics as the optimal solution is approached. We improved the convergence of our algorithm with the conjugate gradient [15], i.e. an estimate of the second-order terms using the results of the previous iterations. Let the step in the decision variables δu_k, and $\delta \Delta t$ be stacked in one vector δp:

$$\delta p = \begin{bmatrix} \delta u_1 & \delta u_2 & \ldots & \delta u_N & \delta \Delta t \end{bmatrix}^T \quad (11)$$

Suppose that the steps in the decision variables of the previous iteration (after clipping) are δp^{old} and the new steps δp^{new} (after clipping). The conjugate gradient direction δp^{CG} is then computed as follows:

$$\delta p^{\text{CG}} = \delta p^{\text{new}} + \frac{\|\delta p^{\text{new}}\|}{\|\delta p^{\text{old}}\|}\delta p^{\text{old}} \tag{12}$$

Because in the first iteration there is no δp^{old}, δp^{CG} is computed in the first iteration as $\delta p^{\text{CG}} = \delta p^{\text{new}}$. The conjugate gradient direction δp^{CG} is the direction that is used in the line search to determine the step-size.

Line-search The line-search chooses the size α of the step that is made in the conjugate gradient direction δp^{CG}. This step-size is not fixed as in the algorithm of Bryson [11], but computed at each iteration. The line search algorithm searches for the step-size that minimizes the value of the objective function of Problem 2. The step-size is restricted to stay within a *trust region* expressed in terms of maximum steps in the controls and the sample time:

$$\begin{aligned} \|\alpha\delta u_k\| &\leq \delta u^{\max} \\ \|\alpha\delta\Delta t_s\| &\leq \delta\Delta t^{\max} \end{aligned} \tag{13}$$

$\delta u^{\max}$ and $\delta\Delta t^{\max}$ are tuning parameters set to $\delta u^{\max} = \frac{1}{2}u^{\max}$ and $\delta\Delta t^{\max} = 0.5\,\text{sec}$. The model equations and the bounds on the decision variables are respected during the line search.

Simulations have made clear that the range of optimal α's is very large. The steps are typically $0.5\Delta u^{\max}$ in the first iterations and become in the order of $10^{-4}\Delta u^{\max}$ close to the optimum. A search is spread over this large range by trying 21 values for α that are "logarithmically equidistant" as follows: if the maximum α that satisfies Equation (13) is denoted $\alpha^{\max}$, 21 values of α are computed by $\alpha_j = \alpha^{\max}\left(10^{-4}\right)^{\frac{j}{20}}$, $j \in \{0, 1, 2, \ldots, 20\}$. This produces 21 values of α where the smallest value equals 10^{-4}. A refined search of again 21 values is then performed in the neighborhood $[\alpha_{l-1}, \alpha_{l+1}]$ of the best α_l. The line-search has a fixed computation time.

Gauss-Newton The Gauss-Newton method [16] estimates the quadratic terms more accurately than the conjugate gradient method, because it exploits the specific structure of problem 2. The objective J_{discrete} in problem 2 is quadratic in the final state, which is a nonlinear function of the decision variables through the model dynamics. At each iteration, the dynamics are linearized along the state trajectory at the previous iteration. The linearizations are obtained by numerical perturbation.

By using the linearized states recursively at each sample instant, the final state is expressed linearly in terms of the decision variables. This approximated final state is inserted into the objective of problem 2, which reduces it to a quadratic problem with bounds on the controls. A small weighting matrix ϵI_{Nm+1} is added to the quadratic terms to prevent an ill-conditioned Hessian, which may lead to numerical difficulties. ϵ is a tuning parameter set to $\epsilon = 1 \cdot 10^{-3}$ and $I_{Nm+1} \in \mathbb{R}^{Nm+1 \times Nm+1}$ is the identity matrix. The resulting quadratic problem is solved with a standard QP-solver and yields the steps in the decision variables δp^{GN}. The line-search is used to determine the step size.

Switching rule The optimization routine starts with the conjugate gradient method. After n_{CG} iterations it switches to the Gauss-Newton method, which is terminated when the steps in the decision variables are smaller than the tolerance value $\|\alpha\delta p^{\text{GN}}\| < \varepsilon$. n_{CG} and ε are tuning parameters set to $n_{\text{CG}} = 3$ and $\varepsilon = 10^{-5}$ respectively. Then the algorithm switches back for n_{cg} times to the conjugate gradient algorithm. If the step-size in all n_{CG} steps is smaller than ε, i.e. $\|\alpha\delta p^{\text{CG}}\| \leq \varepsilon$, the algorithm terminates. Otherwise, the algorithm switches back to the Gauss-Newton method.

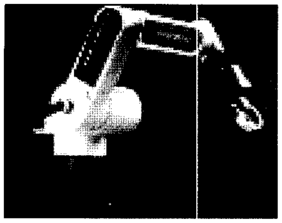
Software implementation We used Autolev[1] to generate the robot model in C-code. The C-code was embedded in a Matlab mex-file. The optimization routine was written in Matlab[2]. For integration of the model equations we used the solver that is built-in Simulink-Matlab, the ode 45 Dormand Prince with variable step-size. Although the line search has a global nature, our algorithm is a local search method.

4 Simulation results

2-DOF Simulations with a 2-DOF robot were performed. Using the same piecewise constant control parametrization, we obtained a final motion time $t_f = 0.389\,\text{sec}$ which is significantly shorter than the time $t_f = 0.671\,\text{sec}$ reported by Dissanayake *et.al.* [6] for the same problem. Because this motion problem is non-singular, a bang-bang control parametrization was used by Dakev *et.al.* [17]. They reported a final motion time of $t_f = 0.387\,\text{sec}$. The very small difference must be attributed to our control parametrization scheme, which does not allow bang-bang controls with arbitrary switching times.

[1] OnLine Dynamics Inc.
[2] The Mathworks, Inc.

5-DOF The method is applied to a simulation model of a 5-DOF robot. The Denavit-Hartenberg parameters are shown in Table 1. These parameters relate to the Eshed[3] MK2 industrial robot.



The Eshed Robotec MK2 robot.

The MK2 is a heavily geared robot resulting in almost decoupled and linear dynamics. The model used in this paper however assumes the MK2 to be actuated by direct drive motors. In terms of energy efficiency and speed of the robot direct drive motors are highly preferable. Moreover the highly nonlinear nature of the associated robot model makes the time-optimal control problem much more difficult and challenging. The masses of the links are located at the center of the link. Tables 2 and 3 give the dynamic parameters. The bounds on the torques are symmetric: $\tau_i^{\min} = -\tau_i^{\max}$. 20 control intervals are used.

The controls, angular positions and velocities of the solution are shown in Figure 1, 2 and 3 respectively. The final position is reached in $t_f = 0.56\,\mathrm{sec}$. Figure 1 reveals that at every time instant there is at least one control in saturation. This is in accordance with the results of Chen and Desrochers [18]. As can be observed from Figure 1, due to the control parametrization the control trajectories are quite smooth instead of exhibiting many switches between the upper and lower bound, which may inflict damage to the robot. The initial guess of the controls and sample time were set to $u_k^{\text{initial}} = 0$, $k \in \{1, 2, \dots, 20\}$ and $\Delta t^{\text{initial}} = \Delta t^{\min}$ respectively. Even though the initial guess is far from the optimum, the algorithm converged well. We set the bounds on the motion time $\Delta t^{\min} = 0.005\,\mathrm{sec}$ and $\Delta t^{\max} = 0.25\,\mathrm{sec}$. These bounds appeared to be wide enough, since the optimal value is $\Delta t = \frac{t_f}{N} = 0.028\,\mathrm{sec}$.

Computation time The computation time for the 5-DOF robot is 21 min. Except for the algorithms in Dubowsky et.al. [9] and Shiller and Dubowsky [10], which both rely on state parametrization, unfortunately all the papers in our reference list do not report computation times. For off-line computation, e.g. for optimal control+LQG design [13], a computation of 21 min is no problem. The computation time is sensitive to lower and upper bounds of the sampling time. The computation times are 32 sec and 4 min for narrow bounds on the sample time of the 2-DOF robot ($\Delta t^{\min} = 0.0385$, $\Delta t^{\max} = 0.04$) and 5-DOF robot ($\Delta t^{\min} = 0.0275$, $\Delta t^{\max} = 0.0285$) respectively.

[3]Eshed robotec, Inc.

Table 1: 5-DOF: Denavit-Hartenberg parameters.

i	$\alpha_i\,[-]$	$a_{i-1}\,[m]$	$d_i\,[m]$	$\theta_i\,[-]$
1	0°	0	0	u_1
2	90°	0.2	0	u_2
3	0°	0.27	0	u_3
4	0°	0.23	0	u_4
5	90°	0.15	0	u_5

Table 2: 5-DOF: masses, inertia and maximum torques.

i	$m_i\,[kg]$	$I_i^{xx}\,[m^2kg]$	$I_i^{yy}\,[m^2kg]$	$I_i^{zz}\,[m^2kg]$
1	2	–	–	0.333
2	1	0.0022	0.0088	0.0094
3	0.5	0.0274	0.0297	0.0037
4	0.3	0.0003	0.0004	0.0006
5	0.1	0.0057	0.0057	0.0007

5 Conclusions

We presented a new algorithm for the computation of bang-bang and singular solutions of the time-optimal control problem for >2-DOF robots with viscous friction and gravity. The algorithm is a combination of a conjugate gradient method and a Gauss-Newton method. In the case of bang-bang solutions the algorithm computed final motion times for a 2-DOF robot, which are only slightly longer due to the control parametrization. As opposed to other algorithms suitable for >2DOF-robots, our algorithm does not suffer from errors introduced by state parametrization. Our algorithm calculated smooth controls on the partially singular 5-DOF robot motion problem. More smooth control trajectories can be obtained by increasing the number of control intervals N. The method is not limited to application on robots but can be applied to other nonlinear optimal control problems with free final time and bounded control. In future work we intend to include obstacle avoidance. Within this method this can be achieved using penalty functions.

Table 3: 5-DOF: initial and final positions, maximum torques and friction coefficients.

i	$\theta^{initial}$ [deg]	θ^{final} [deg]	$T_i^{\max}\,[N]$	$K(i,i)\,[\frac{Nm}{s}]$
1	0	90	15	0.5
2	0	−90	10	0.5
3	0	90	5	0.5
4	0	0	5	0.5
5	0	0	5	0.5

References

[1] R. Fotouhi and W. Szyszkowski, "Further results on time-optimal control of two-link rigid manipulators," *International journal of robotics and automation*, vol. 13, no. 4, pp. 113–124, 1998.

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

[3] L. G. van Willigenburg and R. P. H. Loop, "Computation of time-optimal controls applied to rigid manipulators with friction," *International journal of control*, vol. 54, no. 5, pp. 1097–1117, 1991.

[4] Y. Chen and J. Huang, "An improved computation of time-optimal control trajectory for robotic point-to-point motion," *International Journal of Control*, vol. 65, pp. 177–194, 1996.

[5] H. P. Geering, L. Guzella, S. A. R. Hepner, and C. H. Onder, "Time-optimal motions of robots in assembly tasks," *IEEE transactions on automatic control*, vol. AC-31, no. 6, pp. 512–518, 1986.

[6] M. W. M. G. Dissanayake, C. J. Goh, and N. Phan-Thien, "Time-optimal trajectories for robot manipulators," *Robotica*, vol. 9, pp. 131–138, 1991.

[7] S. F. P. Saramago, "Optimization of the trajectory planning of robot manipulators taking into account the dynamics of the system," *Mechanical machine theory*, vol. 33, pp. 883–894, 1998.

[8] G. Fang and M. W. M. G. Dissanayake, "A neural network-based method for time-optimal trajectory planning," *Robotica*, vol. 16, pp. 143–158, 1998.

[9] S. Dubowsky, M. A. Norris, and Z. Shiller, "The optimal trajectory planning for robotic manipulators with obstacle avoidance: A CAD approach," in *Proceedings of the IEEE International Conference on Robotics and Automation*, (San Francisco), pp. 1906–1912, 1986.

[10] Z. Shiller and S. Dubowsky, "On computing the global-optimal motions of robotic manipulators in the presence of obstacles," *IEEE transactions of robotics and automation*, vol. 7, no. 6, pp. 785–797, 1991.

[11] A. E. Bryson, *Dynamic Optimization*. Menlo Park, California: Addison Wesley, 1999.

[12] J. J. Craig, *Introduction to Robotics*. Massachusetts: Addison-Wesley, 1989.

[13] L. G. van Willigenburg, "Computation and implementation of digital time-optimal feedback controllers for an industrial x-y robot subjected to path, torque and velocity constraints," *International journal of robotics research*, vol. 12, pp. 420–433, 1993.

[14] A. Weinreb and A. E. Bryson, "Optimal control of systems with hard control bounds," *IEEE transactions on automatic control*, vol. AC-30, no. 11, pp. 1135–1138, 1985.

[15] A. P. Sage and C. C. White, *Optimum Systems Control*. New Jersey: Prentice-Hall, 1977.

[16] R. Fletcher, *Practical Methods of Optimization*. New York: John Wiley and Sons, 1980.

[17] N. V. Dakev, A. J. Chipperfield, J. F. Whidborne, and P. J. Fleming, "An evolutionary algorithm approach for solving optimal control problems," in *IFAC 13th Triennial World Congress, San Fransisco*, pp. 321–326, IFAC, 1996.

[18] Y. Chen and A. A. Desrochers, "A proof of the structure of the minimum-time control law of robotic manipulators using a hamiltonian formulation," *IEEE transactions on robotics and automation*, vol. 6, no. 3, pp. 338–393, 1990.

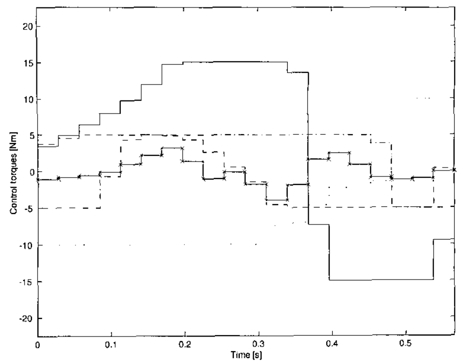


Figure 1: Controls of 5–DOF robot: axis 1: — , axis 2: · · ·, axis 3: –· axis 4: – –, axis 5: -x-.

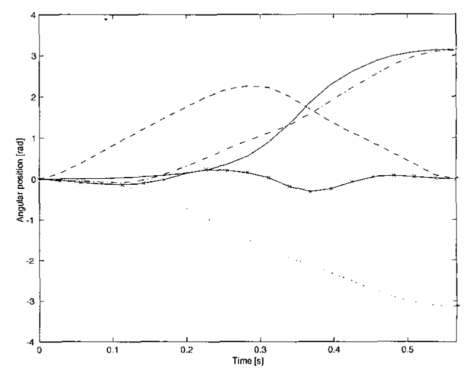


Figure 2: Joint positions of motion of 5-DOF robot: axis 1: — , axis 2: · · ·, axis 3: –· axis 4: – –, axis 5: -x-.

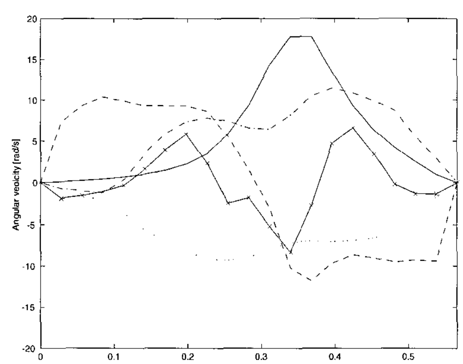


Figure 3: Joint velocities of motion of 5-DOF robot: axis 1: — , axis 2: · · ·, axis 3: –· axis 4: – –, axis 5: -x-.