Robotica (2000) volume 18, pp. 423–428. Printed in the United Kingdom © 2000 Cambridge University Press

Optimal trajectory planning and sliding mode control for robots using evolution strategy

Young-Kiu Choi,* Jin-Hyun Park,** Hyun-Sik Kim,† and Jung Hwan Kim‡

(Received in Final Form: September 7, 1999)

SUMMARY

Although robots have some kinematic and dynamic constraints such as the limits of the position, velocity, acceleration, jerk, and torque, they should move as fast as possible to increase the productivity. Researches on the minimum-time trajectory planning and control based on the dynamic constraints assume the availability of full dynamics of robots. However, the dynamic equation of robot may not often be exactly known. In this case, the kinematic approach for the minimum-time trajectory planning is more meaningful. We also have to construct a controller to track precisely the minimum-time trajectory. But, finding a proper controller is also difficult if we do not know the explicit dynamic equations of a robot.

This paper describes an optimization of trajectory planning based on a kinematic approach using the evolution strategy (ES), as well as an optimization of a sliding mode tracking controller using ES for a robot without dynamic equations.

KEYWORDS: Time-optimal trajectory; Evolution strategy; Sliding mode controller.

1. INTRODUCTION

Robot manipulators have become indispensable for factory automation. In order to increase productivity, it is desirable to move robots as fast as possible. However, robots have some physical constraints such as the limits of the position, velocity, acceleration, jerk, and torque. The problem may be formulated as a minimum-time control problem that is very difficult to solve because of the nonlinearity and couplings in robot dynamics.[1] The minimum-time control problem can be split into two subproblems: the minimum-time trajectory planning and the trajectory tracking. The trajectory planner provides the trajectory tracker with the control commands such as the desired values of a robot's position and velocity, while the tracker is responsible for the robot to follow the control commands as closely as possible.[2]

* Dept. of Electrical Engineering, Pusan National University, Pusan 609–735 (S. Korea)
E-mail: ykichoi@hanmail.net
** Dept of Industrial Automation Engineering, Chinju National University (S. Korea)
† Agency for Defence Development (S. Korea)
‡ Centre for Advanced Computer Studies, University of Southwestern Louisiana, Lafayette, LA 70504 (USA)

Lin, Chang, and Luh[3] proposed a minimum-time trajectory planning method with kinematic constraints on the position, velocity, acceleration, and jerk. The desired path was given through some finite knot points, and the minimum-time trajectories connecting the knot points were investigated with the flexible polyhedron search algorithm. Bobrow, Dubowsky and Gibson,[4] Shin and McKay,[2,5] and J.-J.E. Slotine and H.S. Yang[6] developed dynamic approaches to find the minimum-time trajectories along a prescribed path under the actuator constraints such as torques, assuming the robot full dynamics are available. Their methods were extended to find the minimum-time paths.[7,8] The research for the minimum-time trajectory planning was also conducted for two robots system.[9] Trajectory tracking along the minimum-time trajectories based on the dynamic approaches is trivial because the exact control torque can be computed from the robot dynamic equation.

Dynamic approaches for finding the minimum-time trajectories and paths are justified if the full dynamic equations of motion for robots are exactly known. In real situations, however, it is often difficult to obtain accurate dynamic equations, and the dynamics are also changeable due to variable payloads. Without the accurate dynamic equations, the dynamic approaches for the minimum-time trajectories and paths may lose their meaning, and we are unable to compute the control torque to drive the robots along the planned trajectories. In this case, the kinematic approaches may be more appropriate than the dynamic ones since the former need not know the robot dynamic equations.

In this paper, we discuss the problem of the minimum-time trajectories and the control strategy to drive the robots along the trajectories, assuming that exact dynamic equations of robots are unavailable. Finite knot points from the path are specified to describe the desired path. The trajectory between the knot points is described by cubic polynomials and parameterized by time intervals between the knot points. Each time interval should be optimized so that the total traveling time is minimized under the constraints. To find the best time intervals, Lin[3] used the flexible polyhedron search method. But this algorithm suffers from characteristics of the local search whose performance depends on initial conditions.[10] In this paper we optimize the time intervals by using the evolution strategy (ES)[11,12] *i.e.* a kind of an evolutionary algorithm that is known to be simple and has an excellent global search feature. To track precisely the minimum-time trajectory, we adopt a sliding mode controller that is a kind

https://doi.org/10.1017/S0263574799002118 Published online by Cambridge University Press

of a robust controller.[13–15] The sliding mode controller can be easily implemented. Due to the assumption on the lack of full knowledge of the robot dynamics, we have to tune iteratively the control parameters such as the sliding surface gradient, saturation function boundary and switching gain. These tuning operations are done successfully through experiments, using ES. Experimental results for the SCARA type robot show a drastic reduction in traveling time of the planned trajectories and an increasing accuracy for tracking the planned trajectories as the search operations of ES are repeated.

2. TRAJECTORY OPTIMIZATION FOR ROBOT MANIPULATOR

2.1 Trajectory construction

The robot task planner generates the ordered sequence of knot points in the Cartesian space, and these knot points are transformed to those in the joint space through inverse kinematics. Robot trajectories are constructed by a sequence of cubic polynomials that connect the knot points in the joint space.[3]

When the knot points in the joint space are specified by $q_1, q_3, q_4, \ldots, q_{n-3}, q_{n-2}, q_n$, and the time intervals h_i's between the knot points are given, the joint trajectory can be described as follows: Figure 1 shows that h_i is the interval between t_i and t_{i+1}. Let $Q_i(t)$, $\dot{Q}_i(t)$, and $\ddot{Q}_i(t)$ be the position, velocity, and acceleration between the knot points q_i and q_{i+1}, respectively. The v_1, a_1, v_n, and a_n denote the initial velocity, initial acceleration, final velocity, and final acceleration, respectively. Then we have the following equations.

$$Q_i(t)=\frac{\ddot{Q}_i(t_i)}{6h_i}(t_{i+1}-t)^3+\frac{\ddot{Q}_i(t_{i+1})}{6h_i}(t-t_i)^3 + \left[\frac{q_{i+1}}{h_i}-\frac{h_i\ddot{Q}_i(t_{i+1})}{6}\right](t-t_i) + \left[\frac{q_i}{h_i}-\frac{h_i\ddot{Q}_i(t_i)}{6}\right](t_{i+1}-t) \quad (1)$$

where $i=1, 2, \ldots, n-1$.

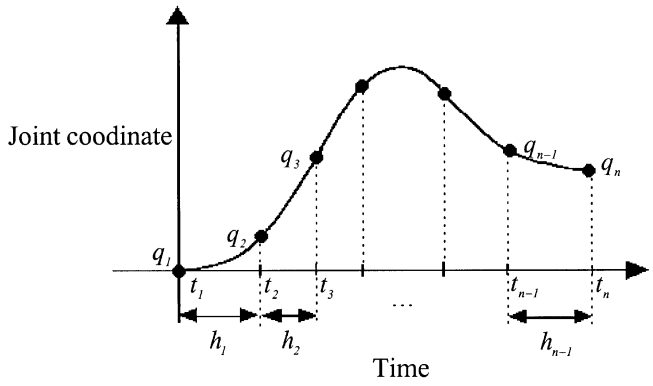


Fig. 1. Robot manipulator trajectory.

$$q_2=q_1+h_1v_1+\frac{h_1^2}{3}a_1+\frac{h_1^2}{6}\ddot{Q}_1(t_2)$$

$$q_{n-1}=q_n-h_{n-1}v_n+\frac{h_{n-1}^2}{3}a_n+\frac{h_{n-1}^2}{6}\ddot{Q}_{n-2}(t_{n-1}).$$

The q_2 and q_{n-1} are determined by the above equations.

$$\dot{Q}_i(t)=\frac{-\ddot{Q}_i(t_i)}{2h_i}(t_{i+1}-t)^2+\frac{\ddot{Q}_i(t_{i+1})}{2h_i}(t-t_i)^2 + \left(\frac{q_{i+1}}{h_i}-\frac{h_i\ddot{Q}_i(t_{i+1})}{6}\right) - \left(\frac{q_i}{h_i}-\frac{h_i\ddot{Q}_i(t_i)}{6}\right) \quad (2)$$

where $i=1, 2, \ldots, n-1$.

$$\ddot{Q}_i(t)=\frac{t_{i+1}-1}{h_i}\ddot{Q}_i(t_i)+\frac{t-t_i}{h_i}\ddot{Q}_i(t_{i+1}) \quad (3)$$

where $i=1, 2, \ldots, n-1$.

The accelerations in each time interval can be determined using the positions of knot points and time intervals, given the velocities and accelerations at the initial and final knot points.

As shown in the above equations, the time interal h_i plays an important role about the velocity and acceleration of the trajectory. Shorter h_i causes higher velocity and acceleration. Then, the problem is to find h_i's such that the total traveling time, the sum of all h_i's, should be minimized under the limits of velocity, acceleration, and jerk. Jerk means the derivative of acceleration with respect to time. Flexible polyhedron search method was used to find the minimum-time trajectory.[3] But the search algorithm is a kind of a local search, and its results largely depend on initial conditions.

In this paper, to find optimal time intervals h_i's we use ES that is simple and has excellent global search performance.

2.2. Evolution strategy

Evolutionary algorithms (EAs), whose studies began in the late 1960s, have prominent capability for optimization. EAs imitate the principle of natural evolution and do not require the continuities and derivatives of objective functions, which are indispensable information to the conventional hill climbing methods. EAs are global search algorithms so that they can have a low probability of local convergence.[11]

Evolution strategy (ES), which is known as one of EAs, has both features of global and local searches, and its search space is influenced by the standard deviation of the random variable.

ES is relatively simple in comparison with other evolutionary algorithms and uses the search parameter itself. The mutation operation of ES is given as

https://doi.org/10.1017/S0263574799002118 Published online by Cambridge University Press

$$x' = x + N(0, \sigma) \tag{4}$$

where x is a parent parameter vector, x' is an offspring parameter vector, and $N(0, \sigma)$ denotes the normal distribution with zero mean and the standard deviation σ. The parent denotes a parameter before operation, and the offspring means a parameter after operation. The σ plays an important role in searching operations. If σ is large, ES seems to be a global search algorithm. If σ is small, ES looks like a local search algorithm. It is usual to have a large σ in the initial phase of search and to have a small σ in the final phase of search.

Time intervals h_i's are iteratively updated to minimize the traveling time under the constraints of velocity, acceleration, and jerk as follows:

$$h_{i,\,offspring} = h_{i,\,parent} + N(0, \sigma'),\ i = 1, 2, \ldots, n-1 \tag{5}$$

where σ' is a standard deviation for the trajectory search.

A fitness function is necessary to select excellent individuals. The fitness function f_p has been chosen as

$$f_p = \frac{1}{\sum_{i=1}^{n-1} h_i + p_{vel} + p_{acl} + p_{jerk}} \tag{6}$$

where

$$p_{vel} = W_{vel} * \left(\sum_{j=1}^{L} \sum_{i=1}^{n-1} OVC_j(i) \right)$$

$$p_{acl} = W_{acl} * \left(\sum_{j=1}^{L} \sum_{i=1}^{n-1} OAC_j(i) \right)$$

$$p_{jerk} = W_{jerk} * \left(\sum_{j=1}^{L} \sum_{i=1}^{n-1} OJC_j(i) \right)$$

W_{vel}, W_{acl}, and W_{jerk} denote penalty weights. L and n mean the number of joints and knot points, respectively. $OVC_j(i)$, $OAC_j(i)$, and $OJC_j(i)$ represent penalties occured by the constraints in the velocity, acceleration, and jerk at the j-th joint in the time interval h_i, respectively. $OVC_j(i)$ can be given as

$$OVC_j(i) = 1 \text{ if } \dot{Q}_i(t) \geq VC_i \text{ for the j-th joint and any } t \in [t_i, t_{i+1}] \tag{7}$$

where VC_i is the velocity constraint. $OAC_j(i)$ and $OJC_j(i)$ can be defined in the same way.

3. TRAJECTORY CONTROL FOR A ROBOT MANIPULATOR

In case when the robot dynamics are exactly known, we can calculate the control torque which drives the robot along the optimized trajectory determined in the trajectory planning phase. However, we often know only the nominal dynamics of robots in the actual situation. If we apply the nominal torque computed from the nominal dynamic equations, it is very hard to obtain a good tracking performance since the nominal torque is a little different from the actual desired torque. In order to compensate for the tracking error, we should add a feedback control loop as shown in Figure 2. T_d means the nominal torque computed with the nominal dynamics of robot along the optimized trajectory. The incremental feedback control u compensates for the error between θ_d and θ. Thus, the total control input to the robot manipulator consists of T_d and u.

We have adopted the sliding mode controller (SMC) as the feedback loop controller. SMC gives us robot control systems that are insensitive to the variation of plant parameters and loads.[13] Researches of SMC began in the USSR and Yugoslavia in the 1960s. SMC has robust control performances with sliding mode actions. Using the discontinuous control input which make the state vector directed towards the sliding surfaces, we can have a fast convergence property along the sliding surface to the origin.[14] The performance of SMC systems is largely determined by some control parameters such as the sliding surface gradient, saturation function boundary layer and switching gain. The sliding surface gradient directly affects the dynamic behaviour. The saturation function boundary layer is used to reduce the chattering problem caused by the discontinuous control input. The switching gain is related to the magnitude of the discontinuous control input that generates the sliding mode actions.[14]

For the precise tracking performances, ES is applied to optimize the sliding surface gradient, saturation function boundary layer and switching gain. The sliding mode controller used in this paper has a dead zone within the saturation function boundary layer to compensate for the

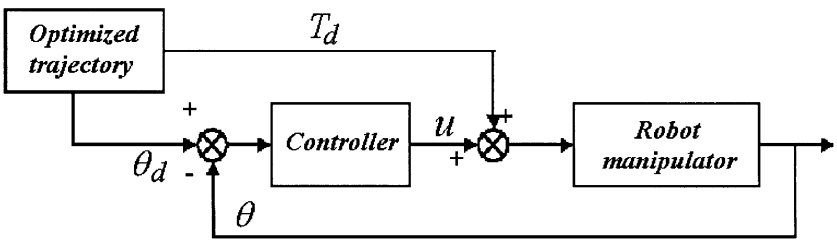


Fig. 2. Control block diagram of robot manipulator.

https://doi.org/10.1017/S0263574799002118 Published online by Cambridge University Press

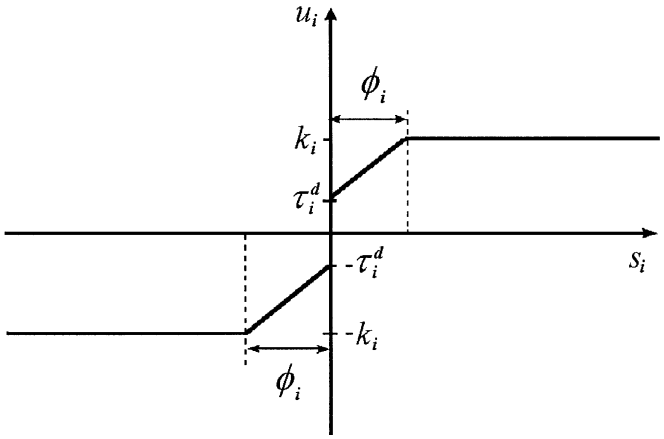


Fig. 3. Sliding mode controller with deadzone torque.

friction forces. Using the sliding mode controller with the dead zone, we have effectively reduced tracking errors for the robot manipulator.

For each joint, the feedback control in Figure 2 is given by

$$u_i = \begin{cases} k_i, & \text{if } s_i > \phi_i \\ -k_i, & \text{if } s_i > -\phi_i \\ \dfrac{(k_i - \tau_i^d)}{\phi_i} s_i + \mathrm{sgn}(s_i) * \tau_i^d, & \text{if } -\phi_i < s_i < \phi_i,\ s_i \neq 0 \\ 0, & \text{if } s_i = 0 \end{cases} \quad (8)$$

where u_i is the control of the i-th joint, and $s_i = c_i x_{i1} + x_{i2} x_{i1}$ is the velocity error of the i-th joint, x_{i2} is the position error of the i-th joint, and c_i means the sliding surface gradient. ϕ_i denotes the saturation function boundary layer, k_i is the switching gain, and τ_i^d is a deadzone torque.

Figure 3 shows the above sliding mode controller with the deadzone torque. The sliding surface gradient c_i, switching gain k_i, and saturation function boundary layer ϕ_i are iteratively updated to reduce tracking errors by using ES as follows:

$$[c_i, k_i, \phi_i]_{parent} = [c_i, k_i, \phi_i]_{offspring} + \mathbf{W}_i * N(0, \sigma_i^c) \quad (9)$$

where $\mathbf{W_i} = [W_i^c, W_i^k, W_i^\phi]$ is a weighting vector, and σ_i^c is a standard deviation for the control parameter search.

The fitness function is necessary to select excellent individuals. The fitness function f_c is defined as

$$f_c = \frac{1}{K_E \sum_{j=1}^{L} \sum_{i=1}^{N} |x_{j2}(iT_s)| + K_C \sum_{i=1}^{L} chat_i + 1} \quad (10)$$

where L is the number of robot joints, T_s is the sampling time, and N is the total number of time steps. K_E and K_C are weighting factors, and $x_{j2}(iT_s)$ is the position tracking error of the j-th joint at the i-th time step. The $chat_i$ is defined as

$$chat_i = \sum_{j=1}^{N} \mathbf{1}\left(\left| \frac{u_i(jT_s) - u_i((j-1)T_s)}{T_s} \right| - \dot{U}_j^{max} \right) \quad (11)$$

Table I. Constraints of the robot

Coordinate	Joint	
Constraints	Joint 1	Joint 2
Velocity	150°/s	150°/s
Acceleration	1200°/s^2	1200°/s^2
Jerk	7000°/s^3	9000°/s^3

where $\dot{U}_j^{max}$ is the maximum changing rate of input torque for the j-th joint, and **1**(x) is a step function which is 1 for a positive x and 0 for a negative x.

4. EXPERIMENTAL RESULTS

We are now ready to apply the above mentioned methods to a real robot manipulator. For experiments, we have chosen the SCARA type robot; only the first 2 joints are used for simplicity. The robot system is composed of the SCARA type robot, a DSP system, and a PC. The constraints are given in Table I. The knot points q_1, q_3, q_4, q_5, q_6, and q_8 are given as (180°, −45°), (162°, −27°), (144° − 9°), (126°, 9°), (108°, 27°), and (90°, 45°), respectively.

ES is used in order to optimize the time intervals $h_i's$. The population size is 21, and the fitness function is the same as that defined in (6). The weighting factors W_{vel}, W_{acl}, and W_{jerk} are assigned to be 1, 1000, and 100, respectively. Figure 4 shows that we can reduce the traveling time of the trajectories for the joint 1 as the number of searches is increased. If we choose the 300-th trajectory as the minimum-time trajectory, we have the minimum traveling time 0.94 seconds that is less than a quarter of 3.98 seconds of the 0-th, initial, trajectory. Therefore, after 300 generations, we can reduce the trajectory traveling time about 4 times. Figures 5, 6, and 7 indicate that the near minimum-time trajectories do not violate the given constraints. For the joint 2, we have results similar to those of the joint 1. The performance comparison of ES and the flexible polyhedron search method (FPS) is depicted in Figure 8. We have 10 trials from random initial conditions for two search methods, ES and FPS. The least traveling time of ES is 0.94 seconds, while that of FPS is 1.14 seconds. Hence, ES gives better a performance than FPS.

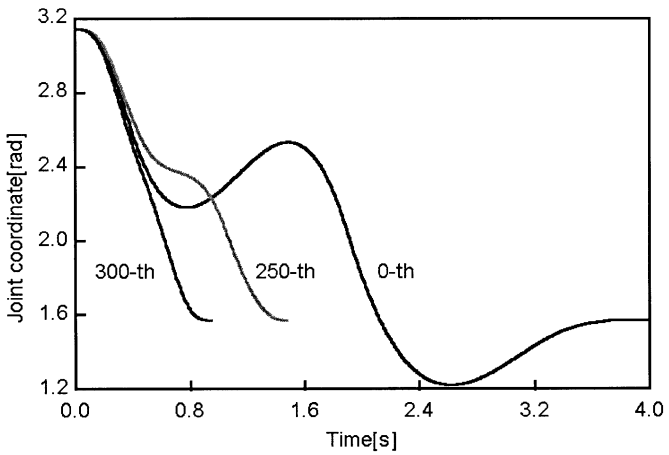


Fig. 4. Reduction of traveling time in position trajectory of the joint 1.

https://doi.org/10.1017/S0263574799002118 Published online by Cambridge University Press

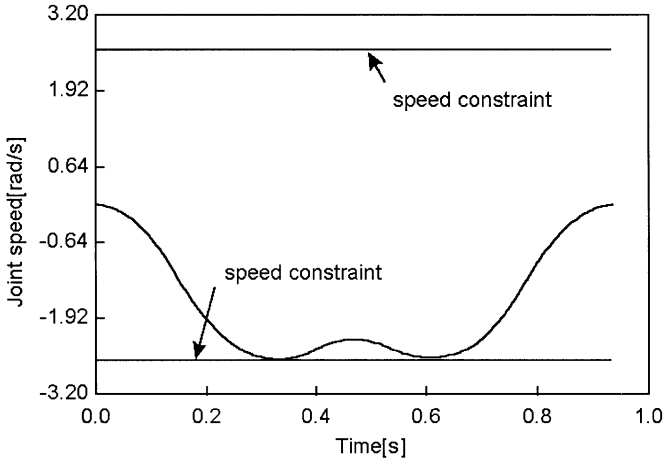


Fig. 5. Optimized speed trajectory of the joint 1.

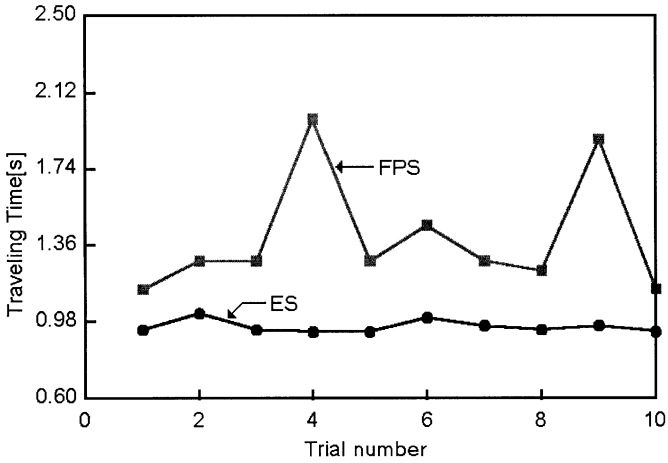


Fig. 8. Performances of ES and FPS.

ES is also used to optimize the control parameters of SMC for the robot. The population size is 6, and the fitness function is the same as that defined in (10). Each of K_C and K_E has 0.03, and the sampling time T_s is 1 millisecond. The total number of steps, N, is given as 940, and the maximum changing rates of input torques $\dot{U}_1^{max}$ and $\dot{U}_2^{max}$ are 0.848 Nm/s and 0.269 Nm/s, respectively. The deadzone torques, τ_1^d and τ_2^d, are 0.034 Nm and 0.011 Nm, respectively. The weighting factors W_1^c, W_2^c, W_1^k, W_2^k, W_1^ϕ, and W_2^ϕ in (9) are 0.05, 0.05, 6, 6, 0.08 and 0.08, respectively. The optimized tracking performance for the sliding mode controller is shown in Figures 9 and 10. The sliding surface gradient, saturation function boundary layer and switching gain of the sliding mode controller are optimized to reduce tracking errors by using ES.

Although large tracking errors exist in the first generation, the errors become reduced as the ES search operation

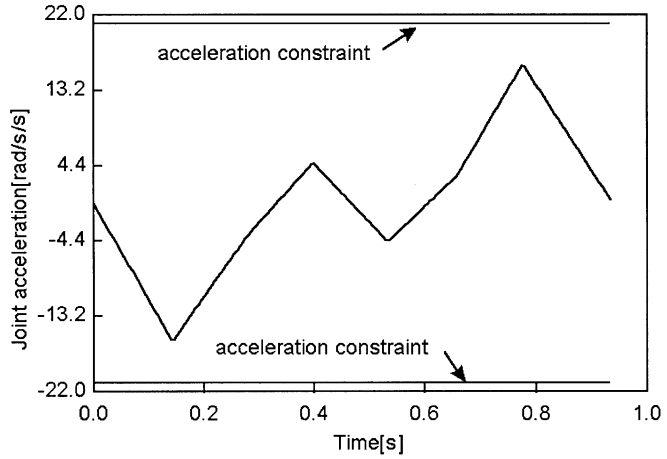


Fig. 6. Optimized acceleration trajectory of the joint 1.

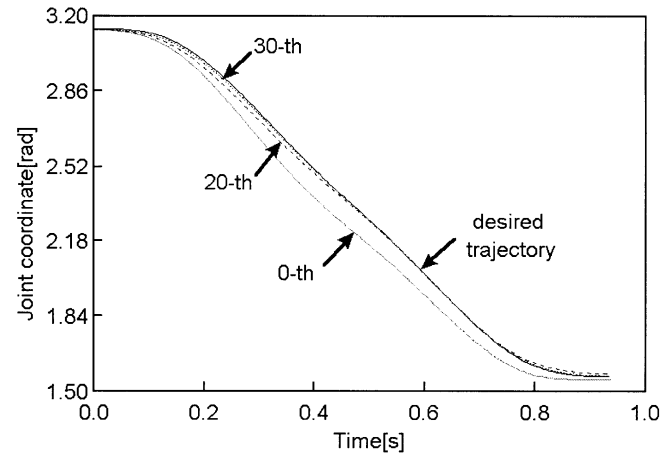


Fig. 9. Position tracking of the joint 1.

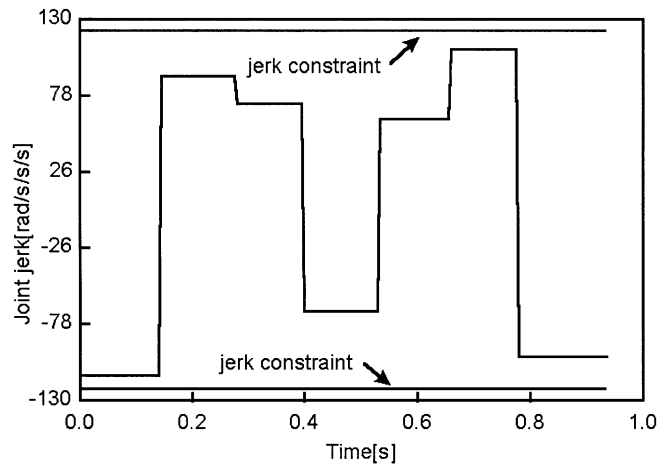


Fig. 7. Optimized jerk trajectory of the joint 1.

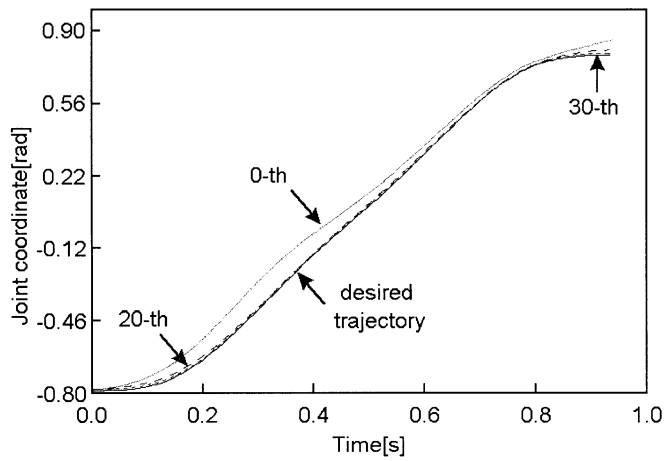


Fig. 10. Position tracking of the joint 2.

https://doi.org/10.1017/S0263574799002118 Published online by Cambridge University Press

is repeated. At the 30-th generation, the maximum tracking errors are 0.21° and 0.18° for joints 1 and 2, respectively. They are so small that we can hardly distinguish the actual trajectory from the preplanned desired trajectory. It takes about 7.3 minutes to conduct experiments for optimization up to the 30-th generation. Therefore, for the repetitive operations such as pick-and-place tasks, our method is useful even though we have not got the exact dynamics of the robots.

5. CONCLUSIONS

In this paper, we discussed the problem of the minimum-time trajectories and the control strategy to drive the robots along the trajectories when the exact dynamic equations of robots are unavailable. The trajectory between the knot points, specified to describe the desired path, is built by cubic polynomials and parameterized by time intervals between the knot points. Each time interval is optimized by use of ES so that the total traveling time can be minimized under the kinematic constraints such as the limits of the position, velocity, acceleration and jerk. We found that ES could get faster trajectories than the conventional methods such as FPS. After achieving the minimum-time trajectory, we adopt a sliding mode controller to track precisely the trajectory. Due to the lack of full knowledge of the robot dynamics, we have iteratively tuned the control parameters using ES. Experimental results for the SCARA type robot show the drastic reduction of traveling time of the planned trajectories and the increasing accuracy for tracking the planned trajectories as the search operations of ES are repeated.

ACKNOWLEDGMENT

The authors acknowledge the help of the Pusan National University whose support made this work possible during the visiting program at the University of Southwestern Louisiana.

References

1. M.E. Khan and B. Roth, "The near-minimum-time control of open-loop articulated kinematic chains," *J. Dynamic Systems Measurement and Control, Transactions of the ASME* **93**, 164–172 (1971).
2. K.G. Shin and N.D. McKay, "Minimum-time control of robotic manipulators with geometric path constraints," *IEEE Trans. Automatic Control* **AC-30**, No 6, 531–541 (1985).
3. C.S. Lin, P.R. Chang and J.Y.S. Luh, "Formulation and Optimization of Cubic Polynomial Joint Trajectories for Industrial Robots," *IEEE Trans. Automatic Control* **AC-21**, 1066–1074 (1983).
4. J.E. Bobrow, S. Dubowsky and J.S. Gibson, "Time-optimal control of robotic manipulators along specified paths," *Int. J. Robotics Research* **4**, No 3, 3–17 (1985).
5. K.G. Shin and N.D. McKay, "A dynamic programming approach to trajectory planning of robotic manipulators," *IEEE Trans. Automatic Control* **AC-31**, No 6, 491–500 (1986).
6. J-J.E. Slotine and H.S. Yang, "Improving the efficiency of time-optimal path-following algorithms," *IEEE Trans. Robotics and Automation* **5**, No 1, 118–124 (1989).
7. K.G. Shin and N.D. McKay, "Selection of near-minimum time geometric paths for robotic manipulators," *IEEE Trans. Automatic Control* **AC-31**, No 6, 501–511 (1986).
8. J.E. Bobrow, "Optimal robot path planning using the minimum-time criterion," *IEEE J. Robotics and Automation* **4**, No 4, 443–450 (1988).
9. J. Lee, "A dynamic programming approach to near minimum-time trajectory planning for two robots," *IEEE Trans. Robotics and Automation* **11**, No 1, 160–164 (1995).
10. D.M. Himmelblau, *Applied Nonlinear Programming* (McGraw-Hill, New York, 1972).
11. T. Back, *Evolutionary Algorithms in Theory and Practice* (Oxford University Press, Oxford 1996).
12. Z. Michalewicz, *Genetic Algorithms + Data Structures = Evolution Programs* (Springer-Verlag, Berlin, 1996).
13. V.I. Utkin, "Variable structure systems with sliding modes," *IEEE Trans. Automatic Control* **AC-22**, No 2, 212–221 (1977).
14. J.Y. Hung, W. Gao, and J.C. Hung, "Variable structure control: a survey," *IEEE Trans. Industrial Electronics* **40**, No 4, 2–22 (1993).
15. W. Gao and J.C. Hung, "Variable structure control of nonlinear systems: a new approach," *IEEE Trans. Industrial Electronics* **40**, No 1, 45–55 (1993).

https://doi.org/10.1017/S0263574799002118 Published online by Cambridge University Press