PERGAMON

Control Engineering Practice 6 (1998) 37–44

Reinforcement learning and its application to force control of an industrial robot

Kai-Tai Song*, Te-Shan Chu

Department of Control Engineering, National Chiao Tung University, 1001 Ta Hsueh Road, Hsinchu 300, Taiwan, R.O.C.

Received 15 September 1996; accepted 11 November 1997

Abstract

This paper presents a learning control design, together with an experimental study for implementing it on an industrial robot working in constrained environments. A new reinforcement learning scheme is proposed, to enable performance optimization in industrial robots. Using this scheme, the learning process is split into generalized and specialized learning phases, increasing the convergence speed and aiding practical implementation. Initial computer simulations were carried out for force tracking control of a two-link robot arm. The results confirmed that even without calculating the inverse kinematics or possessing the relevant environmental information, operating rules for simultaneously controlling the force and velocity of the robot arm can be achieved via repetitive exploration. Furthermore, practical experiments were carried out on an ABB IRB-2000 industrial robot to demonstrate the developed reinforcement-learning scheme for real-world applications. Experimental results verify that the proposed learning algorithm can cope with variations in the contact environment, and achieve performance improvements.

Keywords: Learning control; stochastic reinforcement learning; industrial robots; force tracking control; performance optimization

1. Introduction

Industrial robots have been extensively used for welding, painting and material transfer in automated factories. For more-difficult tasks such as manipulating tools for assembly, grinding and deburring, the robot interacts more closely with its environment, and the inclusion of force information in the control of robot motion is therefore necessary. Fig. 1 illustrates such a working situation, where the normal force and tangential velocity of the tool have to be under simultaneous control for proper operation. Most commercially available robots, however, work solely as position-controlled devices, and have no way of directly controlling the contact forces between tool and workpiece. Increasing the adaptability of industrial robots by developing force controllers has been an active research area (Stepien et al., 1987; De Schutter and Van Brussel, 1988; Song and Li, 1995; Whitcomb et al., 1996). A number of force-control algorithms have been proposed. Almost all of them assume that the dynamic model of the manipulator, as well as the environmental states (shape and stiffness), are known to the designer. This is not always possible in practice, however, especially when the difficulty of estimating environmental variables is taken into account. Considerable time may be required to derive a model that is sufficiently accurate for acceptable performance. Moreover, in many practical applications, contact conditions such as the shape and resilience of the environmental elements are irregular and time-varying. It can be extremely difficult to accomplish force-tracking control using conventional model-based approaches in such situations.

In this paper, reinforcement learning is applied to the force-tracking control of an industrial robot. It is desirable that improved performance of a robot manipulator should be obtained through cyclic practicing. Learning control can achieve improved control performance when system information is unknown or incompletely known. Recent developments in reinforcement learning methods have given rise to their application in the control of intelligent robotic systems (Gullapulli et al., 1994). In this approach, the performance is improved by iterative training. It is therefore employed here to deal with the uncertainties that a robot manipulator faces while interacting

* Corresponding author. Fax: 886-3-5715998; e-mail: ktsong@cc.nctu.edu.tw.

0967-0661/98/$19.00
PII S0967-0661(97)10058-2

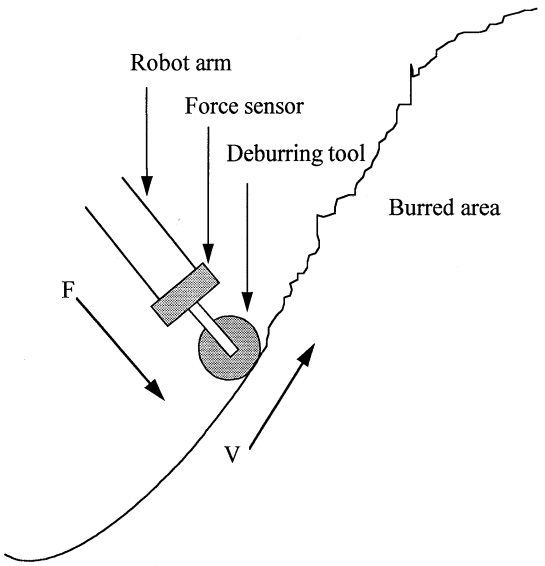

Fig. 1. An illustration of a deburring operation using a robot arm equipped with a force/torque sensor.

with its environment. The approach proposed in this work is to directly adjust the control law through reinforcement learning. It was initially hoped that optimal control rules could be found for a specific task by actively exploring the control space. Via learning, these rules can be found without knowing either the exact robot model or the environmental conditions in which it operates. However, although existing learning algorithms are functionally sufficient, they generally demand so many iterations that the learning process tends to be too slow. Their feasibility for practical application is therefore limited, and the development of more-efficient learning algorithms requires urgent attention. This paper proposes a new reinforcement learning scheme to enable performance optimization in robot force control. Using this scheme, the learning process is split into generalized and specialized learning phases, increasing the convergence speed of learning and aiding practical implementation.

The rest of this paper is organized as follows. Section 2 describes the theoretical background to reinforcement learning. Section 3 presents a new reinforcement learning control method for robotic force-tracking. Computer simulations of this scheme are described in Section 4. In Section 5, a practical realization of the proposed algorithm is presented through force-tracking control of an industrial robot. It demonstrates the feasibility of reinforcement learning in real-world applications. Section 6 presents the conclusions.

2. Background review

In reinforcement learning, a system improves its performance by receiving feedback from its environment in the form of a reward or penalty commensurate with the appropriateness of the system's response. The system then uses this feedback to adapt its behavior so as to maximize the probability of receiving such rewards in the future. Early approaches to reinforcement learning were based on the theory of learning automata (Narendra and Thathachar, 1974). The algorithm selects actions probabilistically from a set of possible actions, and updates its action probabilities on the basis of evaluation feedback. The task is to maximize the expected value of the evaluation received. Although stochastic learning automata have mostly been studied in a non-associative form (that is, where they search for a single optimal action), they can be extended to learn mappings, such as control rules, that associate input patterns with actions. Barto et al. (1981) combined stochastic learning automata with parameter estimation of pattern recognition by parameterizing the mapping from pattern input to action probabilities. As these parameters are adjusted under the influence of evaluation feedback, the action probabilities are adjusted to increase the expected evaluation. Barto et al. termed their network an *associative search network* because it actively sought the optimal output pattern (by a random search) to be associated with each input pattern (associative memory). Barto and Anandan (1985) presented an algorithm of this type, termed the "associative reward-penalty", or A_{R-P}, algorithm, and proved a convergence theory. To solve difficult control problems such as balancing a cart-pole system, Barto et al. (1983) proposed a reinforcement learning scheme using two networks. The learning system consisted of a single adaptive critic element (ACE) and a single associative search element (ASE). The ASE associates the input and output by searching under the influence of reinforcement feedback. The ACE constructs a more informative evaluation function than actual evaluation feedback alone can provide. This two-element cooperative learning mode has been applied to many other learning control systems (Anderson, 1987; Guha and Mathur, 1990; Porcino and Collins, 1990). To increase the learning efficiency for situations where the control objective is achieved after a sequence of control actions (such as in a cart-pole system or chess game), Sutton (1988) proposed an algorithm called the "temporal difference" (TD) algorithm, to predict the evaluation signal for each control action before the terminal state was arrived at.

In the learning automata or the associative reward-penalty algorithm, reinforcement learning can only deal with problems that can be handled by discrete output actions. This is not suitable for most control applications, in which continuous control signals are required. A stochastic reinforcement learning algorithm for learning functions with continuous outputs was proposed by Gullapulli (1990). This algorithm is based on a stochastic real-valued (SRV) unit that contains two elements. As shown in Fig. 2, unit 1 is the learning element that

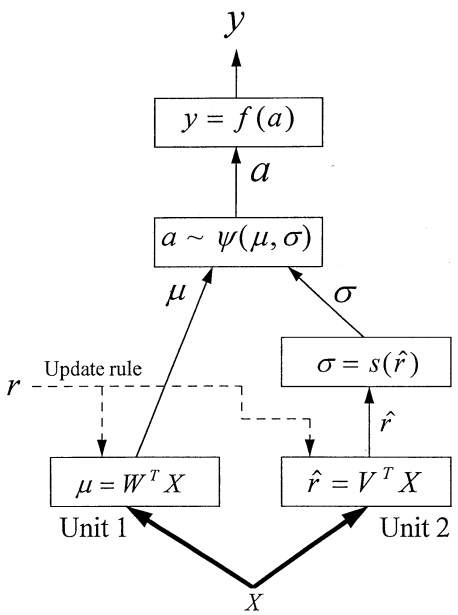

Fig. 2. Stochastic real-valued unit.

produces an output which is a real-valued function of the inputs. Unit 2 is a reinforcement predictor that estimates the expected value of the reinforcement signal. The first unit can be thought of as estimating the mean value μ, and the second as determining the variance σ (by a monotonically increasing, non-negative function s, see Fig. 2). The random search is accomplished by taking the normal distribution $\Psi(\mu, \sigma)$. A random variable a is generated in this process. An output function y then maps a to the control output. The parameters in W and V are adjusted by evaluation feedback from the environment. Because the unit 2 predicts (tracks) the evaluation feedback, the learning rule used for adjusting the parameters in V is usually a least-mean-square (LMS) method (Widrow and Stearns, 1985):

$$\Delta V = \beta(r - \hat{r})X, \tag{1}$$

where X is the input vector, β is the learning rate, r is the evaluation feedback or reinforcement from the environment and $\hat{r}$ is the predicted reinforcement.

The learning rule for adjusting the parameters in W of unit 1 is

$$\Delta W = \alpha(\hat{r} - r)\left(\frac{a - \mu}{\sigma}\right)X, \tag{2}$$

where α is the learning rate parameter. The fraction in (2) can be thought of as normalized noise (Gullapulli, 1990). If noise causes the unit to receive a reinforcement signal that is lower than the predicted value ($\hat{r} - r > 0$), then it indicates the search direction is correct, and that it should update the mean value μ towards a. It is therefore desirable for the unit to have an action closer to the current action. On the other hand, if noise causes the unit to receive a reinforcement signal that is greater than the expected value ($\hat{r} - r < 0$), then it should update its mean output value in the opposite direction.

3. Proposed algorithm

In practice, the learning controller must remember the learned, correct mapping of input and output variables. Two major approaches to associative memory are: the BOXES system and artificial neural networks (ANNs). ANNs have been used for associative memory in many applications. With adequate hidden-layer elements, any non-linear mapping can be accomplished by a three-layered feedforward network. However, the parameters for ANNs are difficult to determine, and many training cycles are generally required to obtain an acceptable generalization property. The BOXES approach was first employed by Michie and Chambers (1986), who partitioned each dimension of the input space into a finite set of intervals. The cross-products of these sets of intervals form a set of boxes in the input space, and a learning element is associated with each box. The inputs are sampled to determine in which box they lie (see Fig. 3). The learning element associated with that box then makes control decisions about the given inputs. The advantage of the boxes approach is its simplicity of implementation, and its drawback is the difficulty of partitioning the input space appropriately. If the partitioning is too rough, the associative property is lost and instability may develop. On the other hand, if the partitioning is too fine, learning will be too slow and will require more memory.

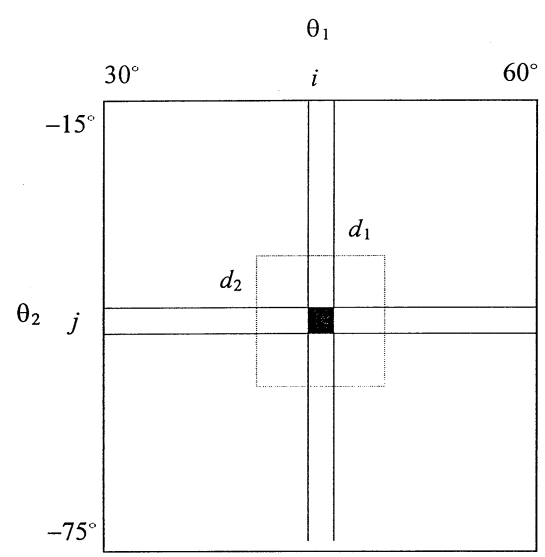

Fig. 3. Generalized learning extent in a 2-DOF BOXES system.

A new learning algorithm is proposed in this paper that combines the advantages of boxes (simplicity) and neural networks (generalization). In this method, learning is split into a generalized learning phase and a specialized learning phase for the association memory using the boxes approach. The generalized learning phase is useful at the very beginning of reinforcement learning, when the system has no learning experience at all. Not only are the parameters of the box in which the inputs belong modified, but also those of the boxes around it. In this way, an input box's *experience* can be extended to related boxes. Fig. 3 illustrates this concept. Specifically, for two input variables, the learning rule for the SRV unit in the generalized learning phase is as follows:

$$\Delta W = \gamma^{\hat{d}_1 + \hat{d}_2}\, \alpha(\hat{r} - r)(a - \mu)X, \tag{3}$$

where γ $(0 < \gamma < 1)$ is the decay rate, $\hat{d}_1 \le d_1$ and $\hat{d}_2 \le d_2$ are the distances from the current box to the input box for the two input variables respectively, and d_1, d_2 are the extent of a neighborhood in a two-dimensional boxes system. Note that in (3) the standard deviation σ is removed. This is because it has no meaning for boxes other than the input box. The learning rule for adjusting the parameters in V is the same as the original equation (1).

Learning is switched to the specialized learning phase when system performance satisfies a preset global criterion. This is determined according to the characteristics of the task and the purpose of control. A good occasion to switch from the generalized to the specialized learning phase is when the system can perform the given task a number of times without failure. During execution, a failure is recorded when the evaluation feedback exceeds the assigned tolerance. After a failure, the system is set to return to the start condition, for a new trial. In the specialized learning phase, as in the original boxes system, only the input box parameters are modified, to fine-tune individual box parameters and give them somewhat specialized characteristics. Fig. 4 depicts the direct reinforcement learning control structure. Exploiting generalized learning, the system can propagate learned experience faster, and thereby decrease the training time. On the other hand, by exploiting specialized learning the system can learn unique features pertaining to each control subspace, and improve the learning convergence. The proposed learning algorithm is summarized as follows:

3.1. Proposed learning algorithm

(S1) Assign initial values to V, W and X.
(S2) Determine the extent of neighborhood d_1, d_2 (if a two-dimensional case).
(S3) Calculate the predicted reinforcement $r(\hat{k})$.
(S4) Calculate control action $y(k)$.
(S5) Obtain system states of the next sample instant $X(k+1)$.

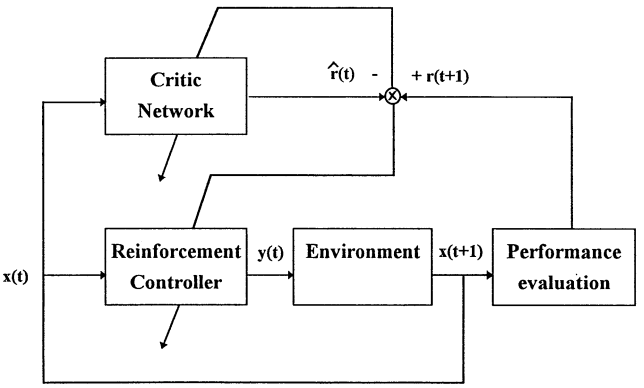

Fig. 4. Direct reinforcement learning control system block diagram.

(S6) Compute the evaluation feedback $r(k+1)$ for $X(k+1)$.
(S7) Update the parameters $V(k)$ in the reinforcement predictor network using (1).
(S8) If the global criterion is not satisfied (i.e., the generalized learning applies), then update the parameters $W(k)$ of the input box using (2) and those of the neighboring boxes using (3); otherwise update the parameters $W(k)$ using (2) (specialized learning).
(S9) If the learning results are acceptable, then stop; otherwise go to S3.

4. Simulation results

Computer simulations of force-tracking control for a two-degree-of-freedom (2-DOF) robot arm were first carried out to verify the proposed reinforcement learning scheme. The lengths of the first and second links of the simulated manipulator were: $l_1 = 1$ m, $l_2 = 0.8$ m respectively. Since most existing industrial robots are position-controlled devices, it was assumed in the simulations that there existed a position controller for motion-command execution. The manipulator dynamics was not taken into account by the learning controller. Therefore, the control outputs were position (angle) commands for each joint. The environment was modelled as a stiffness. This implies that the normal force being applied is proportional to the difference between the environment-surface position and the current position of the end-effector. Under these assumptions, the proposed controller was set to learn both the inverse kinematics of the robot arm and the contact environment stiffness.

In the beginning, the system was in the generalized learning phase. It switched to the specialized learning phase when the robot was able to finish a certain number of trials without failure. The evaluation feedback r functioned as the local criterion, providing simultaneous evaluations of the force and velocity errors to the

learning controller. One simulation tracked a circular surface with a prescribed contact force; this is explained in detail below.

Fig. 5 illustrates the robot arm's initial position and its environment. In this configuration, the arm had an initial contact force $F = 0.7212N$, where the stiffness of the environment was assumed to be 100 N/m. One input variable to the controller θ_1 was partitioned into 100 parts in the range of $45^\circ \pm 15^\circ$, and the other input variable θ_2 was partitioned into 200 parts in the range of $-45^\circ \pm 30^\circ$. This yielded a total of 20 000 boxes for this system. In generalized learning, the decay rate $\gamma = 0.9$. The parameters for specifying the range of updating $d_1 = d_2 = 5$. Therefore, 121 boxes were updated in each iteration. The learning rates α and β were both set to 0.5 for the generalized learning phase. In the specialized learning phase, the learning rates were $\alpha = 0.1$ and $\beta = 0.5$ respectively. In this simulation, the global criterion was set to 10 successive successful trials of the specified trajectory. Here the number 10 was selected by experience. For fewer than 10 trials, the primary learning stage would not be completed. On the other hand, a greater number of trials (in the generalized learning phase) would result in inefficient learning. The evaluation feedback was designed by taking into account that the specified contact force and velocity were 1N and 1cm/step respectively. It was calculated using the following equation:

$$r = \frac{force - error(N)}{2(N)} + \frac{velocity - error(cm)}{2(cm)}. \quad (4)$$

If this value exceeded 1, it constituted a failure and the robot returned to the start position. The trajectory was designed to complete the circular path in 45 steps. Fig. 6 shows the simulation results for the number of steps the robot moved in each repetition. It would not complete the 45-step circular path if the evaluation feedback of any single step was greater than 1. Note that a moving average is used for every 10-set of data in this plot. Fig. 7 shows the learned trajectories of joint 1 and joint 2 respectively. Fig. 8 shows the evolution of the evaluation feedback in this simulation. Fig. 9 shows the force error. In these two figures, the solid lines indicate the final results when the learning converged. The dashed lines are the data taken when the learning process was switched from the generalized learning phase to the

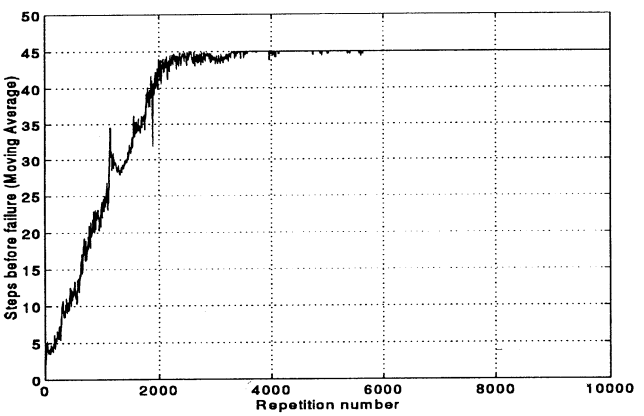

Fig. 6. Simulation result: number of steps the robot moved in each repetition.

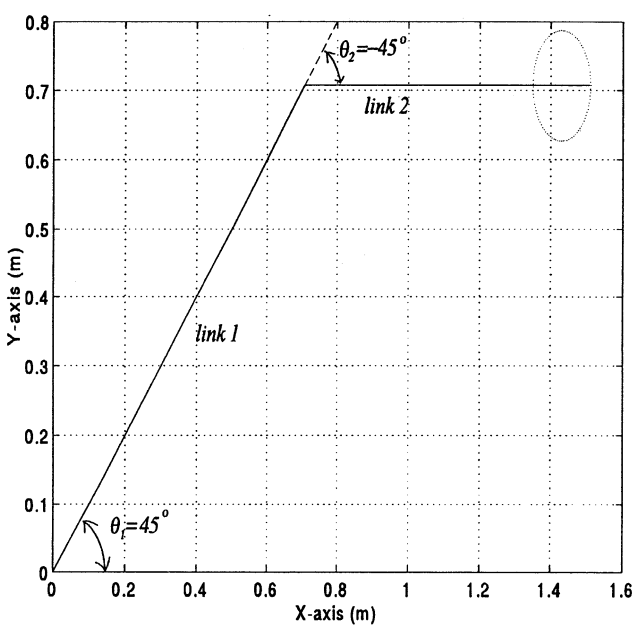

Fig. 5. Initial position of the two-link robot arm in simulation.

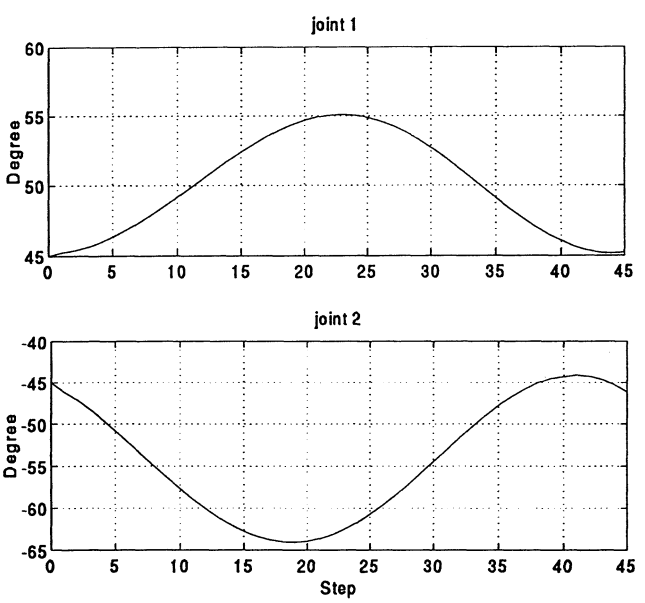

Fig. 7. Simulation result: learned trajectories of joint 1 and joint 2.

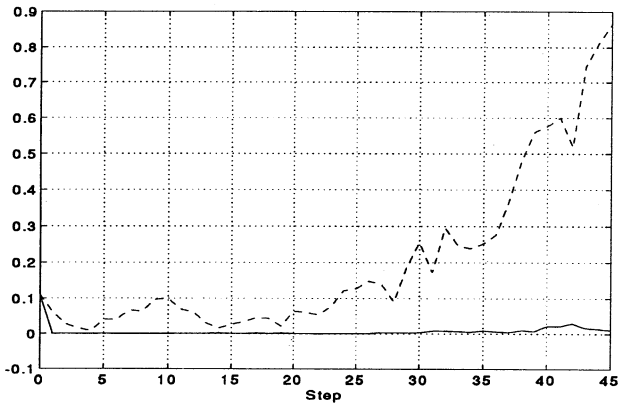

Fig. 8. Simulation result: evaluation feedback.

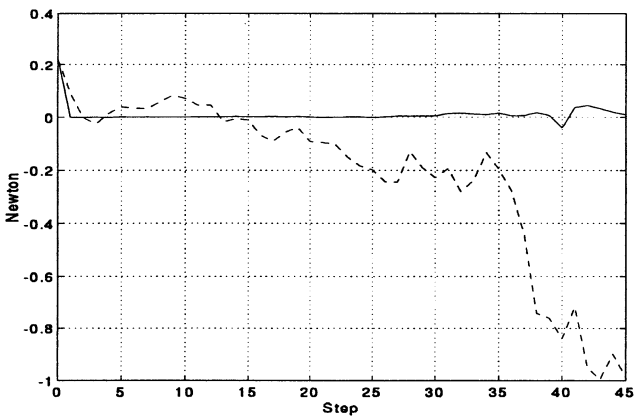

Fig. 9. Simulation result: force error.

specialized learning phase. It can be seen from these figures that the evaluation feedback and force error were still not acceptable, even though the global criterion was satisfied. This was because 10 successful trials of the specified task were adjudged using the tolerance assigned to (4). The learning had not yet converged at this point, and the performance would not be satisfactory. However, in the subsequent specialized learning phase the system improved its performance after more learning iterations. As the learning converged, optimal performance was obtained, as shown by the solid lines. Fig. 10 illustrates the learned trajectory of the simulated manipulator. A dash-lined ellipse represents the contact environment. It can be seen from the figure that the tool tip accomplished a circular path, as desired.

5. Experimental results

An experimental verification was carried out in the laboratory to demonstrate the feasibility of using the proposed scheme for practical force-tracking control. In the experiment, the learning ability was also verified

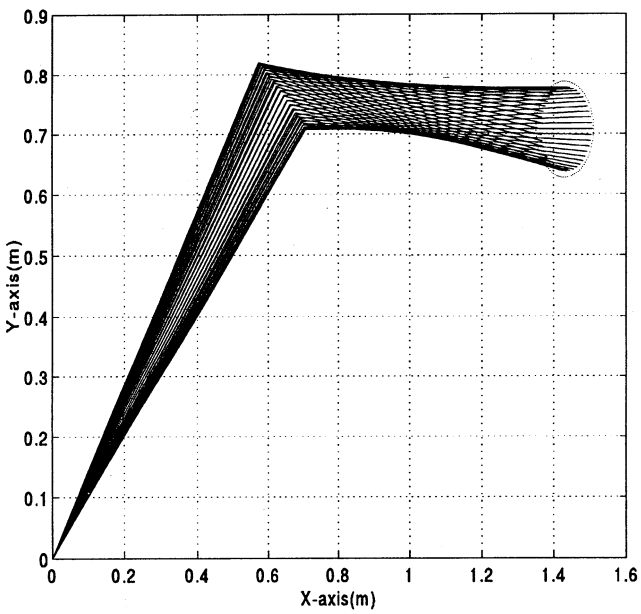

Fig. 10. Simulation result: learned trajectory of the simulated manipulator.

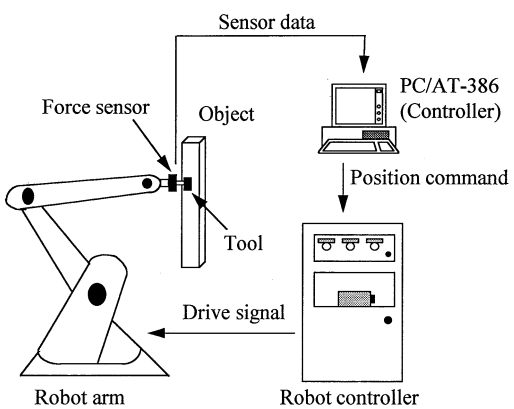

Fig. 11. Experimental setup for force-tracking control.

under the influence of sensor noise and variations in the contact environment. Fig. 11 shows the experimental setup, where an ABB IRB 2000 industrial robot equipped with a JR^3 6-DOF force/torque sensor was used for force-tracking control. Cartesian end-effector position commands were sent to the robot controller via a computer link. The end-effector was a simple roller, which was mounted on the force/torque sensor. The contact surface was made of a sponge-type material. A soft environment was provided to test the learning control algorithm. The normal contact force was obtained from the force/torque sensor, and the reinforcement learning algorithm was run on a PC/AT-386. The experiment was arranged as a single-input-single-output (SISO) system; that is, learning control was applied to one degree of

freedom in Cartesian space. The contact force was controlled through a position-control loop along the Y-axis.

One experimental result is presented here. The desired trajectory of the robot was to move at a constant speed in the Z-axis, with a prescribed contact force of 2 Kg. This trajectory was generated to the manipulator as a position incremental of -5 mm (the negative sign means a downward motion) for 20 steps. In this experiment, $\gamma = 0.8, \alpha = \beta = 0.5$ for the generalized learning phase, and $\alpha = 1, \beta = 0.5$ for the specialized learning phase. Fig. 12 shows the recorded positions along the Y-axis in the reinforcement learning process. Note that in the experiment the contact force was controlled along this axis via the position loop of the manipulator. In this figure, the solid lines are the mean positions (see Fig. 2) and the dashed lines are the random-searched position commands. As learning proceeded, it can be seen from the figure that the search range became smaller and smaller. This was because the controller output approached the ideal output. The error converged to an acceptable level after about 200 learning iterations. Fig. 13 shows the measured contact force at the tool tip in the learning process. In the figure, there are large force errors at the beginning (the first trial). At the 150th repetition (dashed-line), the force error decreases. After 237 iterations the contact force converges to the desired value. Fig. 14 depicts the initial contact force with respect to the repetitions of the learning cycles. Notably, the initial contact force changes. This was caused by variations in the shape and stiffness of the sponge after so many instances of manipulations (rubbing), applied to it in the experiment.

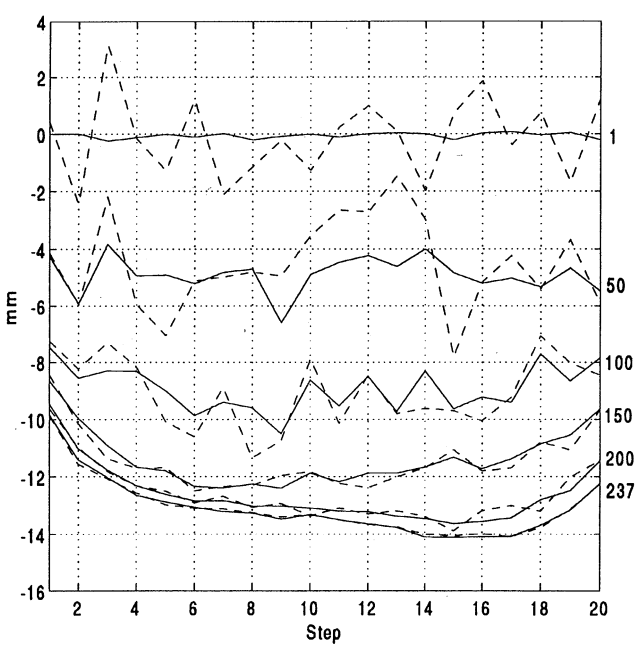

Fig. 12. Experimental results of mean (solid lines) and random-searched position commands (dashed lines) in the learning process.

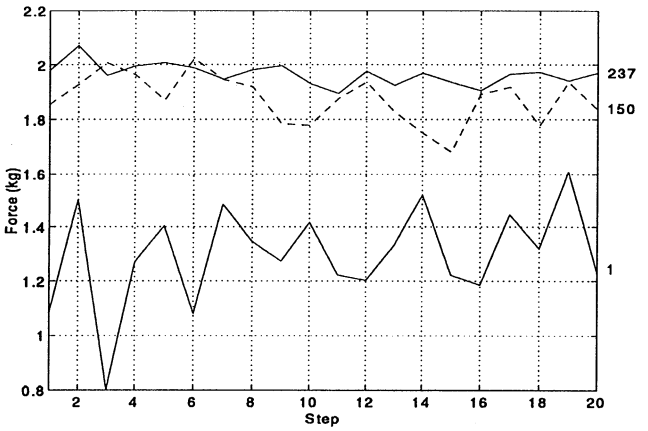

Fig. 13. Experimental results of normal contact force at the tool tip.

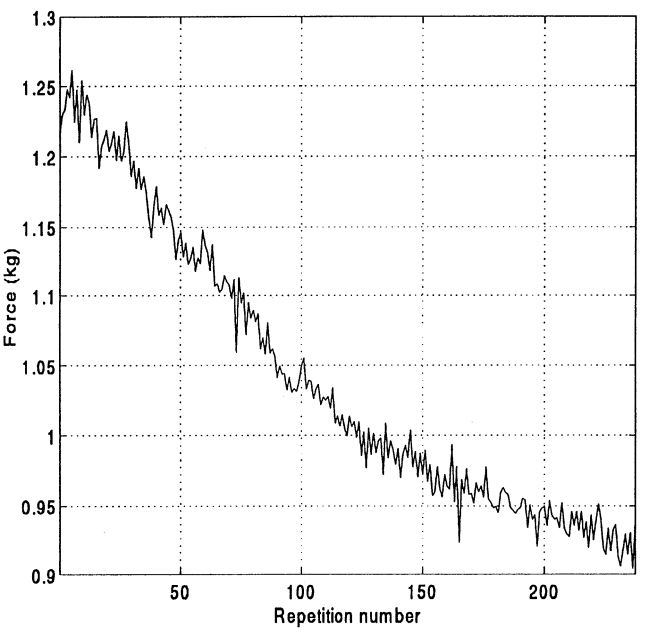

Fig. 14. The change in the initial contact force due to the variations in shape and stiffness of the sponge in the experiment.

However, it can be seen from the force responses in Fig. 13 that this variation in the environment has no influence on the control performance achieved. In fact, the proposed learning algorithm will succeed as long as the learning speed remains faster than the variation rate of the contact environment.

6. Conclusions

In this work, a reinforcement learning control design was developed for robot position and force tracking in time-varying environments. A new learning algorithm is proposed, in which the learning process is split into

a generalized learning phase and a specialized learning phase. This scheme combines the advantages of the conventional boxes system and ANNs for associative memory in the reinforcement learning process. It makes learning more efficient and therefore suitable for practical applications. Simulation results show that the rules that simultaneously control force and velocity can be learned through active exploration, without prior knowledge of either the robot's inverse kinematics or the environment state. The experimental results further demonstrate that although sensor noise existed and variations in the contact surface characteristics occurred, the learning was still successful over 237 repetitions using this algorithm. Extensive studies will be continued in the future to apply reinforcement learning techniques for performance optimization in robotic systems.

Acknowledgement

This work was supported by the National Science Council, Taiwan, Republic of China, under grant number NSC-85-2213-E-009-094.

References

Anderson, C.W., 1987. Strategy learning with multilayer connectionist representation. Proc. of the 4th Int. Workshop on Machine Learning, pp. 103–114.

Barto, A.G., Sutton, R.S., Brouwer P.S., 1981. Associative search network: a reinforcement learning associative memory. Biological Cybernetics, 40, 201–211.

Barto, A.G., Sutton, R.S., Anderson, C.W., 1983. Neuronlike adaptive elements that can solve difficult learning control problems. IEEE Trans. on Sys. Man. and Cyber., Vol. SMC-13, No. 5, pp. 834–846.

Barto, A.G., Anandan, P., 1985. Pattern–recognizing stochastic learning automata. IEEE Trans. Sys. Man. and Cyber., Vol. SMC-15, No. 3, pp. 360–375.

De Schutter, J., Van Brussel, H., 1988. Compliant robot motion II. A control approach based on external control loops, International Journal of Robotics Research, pp. 18–32.

Guha, A., Mathur, A., 1990. Setpoint control based on reinforcement learning. Proc. of IJCNN'90, Vol. II, pp. 511–514.

Gullapulli, V., 1990. A stochastic reinforcement learning algorithm for learning real-valued functions. Neural Networks, 3, 671–692.

Gullapulli, V., Franklin, J.A., Benbrahim, H., 1994. Acquiring robot skills via reinforcement learning. IEEE Control Systems, 14(1), pp. 13–24.

Michie, D., Chambers, R.A., 1986. BOXES: an experiment in adaptive control, Machine Intelligence 2, E. Dale, D. Michie Eds., pp. 137–152.

Narendra, K.S., Thathachar, M.A.L., 1974. Learning automata – a survey, IEEE Trans. on Sys. Man. and Cyber., Vol. 14, pp. 323–334.

Porcino, D.P., Collins, J.S., 1990. An application of neural networks to the guidance of free-swimming submersibles. Proc. of IJCNN'90, Vol. II, pp. 417–420.

Song, K.T., Li, H.P., 1995. Design and experiment of a fuzzy force controller for an industrial robot. In: Proceedings of The National Science Council, R.O.C. Part A: Physical Science and Engineering, pp. 26–36.

Stepien, T.M., Sweet, L.M., Good, M.C., Tomizuka, M., 1987. Control of tool/workpiece contact force with application to robotic deburring. IEEE Journal of Robotics and Automation, Vol. RA-3, No.1, pp. 7–18.

Sutton, R.S., 1988. Learning to predict by the methods of temporal differences. Machine Learning, pp. 9–44.

Whitcomb, L., Arimoto, S., Naniwa, T., Ozaki, F., 1996. Experiments in adaptive model-based force control. IEEE Control Systems, 16(1), pp. 49–57.

Widrow, B., Stearns, S.D., 1985. Adaptive Signal Processing. Prentice-Hall, Englewood Cliffs, NJ.