Proceedings of the 2006 American Control Conference
Minneapolis, Minnesota, USA, June 14-16, 2006

Coordination of Hydraulic Manipulators by Reinforcement Learning

Mark Karpenko, John Anderson and Nariman Sepehri

Abstract— **In this paper, a reinforcement learning method is applied to coordinate a pair of horizontal hydraulic actuators engaged in the cooperative positioning of an object. The goal is to enable the actuators to discover how to intelligently select control actions that tend to reduce the interaction forces directed along the axis of motion, while maintaining the desired trajectory. First, a detailed and realistic dynamic model of the entire system is derived. A multi-layer reinforcement learning neural network control architecture is designed next to regulate the interaction force during positioning. To regulate the interaction force, the neural network measures the interaction force and proposes a modification to the *a priori* prescribed formation constrained position trajectory. Each actuator system is outfitted with such a neural controller so that a decentralized reinforcement learning control system results. Simulations demonstrate the efficacy of the approach towards reducing the interaction forces and minimizing the associated object internal force in a single degree of freedom.**

I. Introduction

One aspect of robotics research that continues to receive much attention in the literature is the manipulation of objects using multi-robot coordinated frameworks. One obvious benefit of such an approach is the ability to manipulate large or awkward objects that would be difficult for a single robot to handle. Another important advantage of using multi-manipulator systems is the possibility of regulating the internal force acting on the object [1]. However, when two or more manipulators are used to move an object, a closed kinematic chain is formed and the motion of one manipulator is translated through the object to affect the motion of the other manipulators [2]. Consequently, it is challenging to develop decentralized control systems for effective coordinated manipulation and much development is needed in this area.

One way to coordinate the efforts of multiple manipulators performing a single task is by the use of reinforcement learning (RL). Reinforcement learning [3] is an artificial intelligence approach for machine learning that adopts the notions of reward and punishment as a means of directing the learning task. Learning by reinforcement is accomplished by trial and error as the learning agents measure the environmental state using sensors and then respond to the current stimuli by executing an action appropriate to the circumstances. By associating a reward with the selected action, it is possible via the RL algorithm to strengthen or weaken the tendency of the agent to execute the same action when a similar state is sensed in the future. The learning goal is therefore to discover an action-selection policy that maximizes the reward. RL has been applied successfully to many difficult control problems (see [4] and [5] for examples). Yet, the application of RL techniques to decentralized control problems, such as coordinating the efforts of multiple manipulators, has only more recently begun to receive attention in the literature.

The literature pertaining to the coordinated control of multiple hydraulic manipulators, in particular, is not well developed and is limited to only a few examples employing centralized control strategies [6], [7], [8]. Moreover, the application of decentralized control techniques to this problem has not been addressed and deserves attention. Due to their widespread use in industry, the development of control strategies for coordinating the efforts of multiple hydraulic manipulators is also of industrial significance. The goal of this paper, therefore, is to employ RL for coordinating a pair of hydraulic actuators performing a single task in a decentralized framework, for the first time.

The coordinated positioning task of interest in this paper is for two hydraulic actuators to rigidly grasp and move an object between two locations along a line while avoiding the application of excessive interaction forces. A single degree-of-freedom manipulation task was chosen to ease the dynamic analysis and maintain the focus of this paper on the RL aspects of the problem. To accomplish the specified positioning task, the hydraulic actuators must follow *a priori* defined formation constrained reference trajectories. Each hydraulic actuator is therefore provided with a simple proportional closed-loop position controller. While it is assumed that the entire manipulation task can be accomplished using this control law alone, positioning errors arise due to the nonlinear nature of the hydraulic functions and imperfect actuator dynamics. These errors result in the application of excessive interaction forces to the manipulated object.

To improve performance, and reduce the interaction force, each hydraulic actuator is also outfitted with a specially designed RL multi-layer neural network controller that can modify the prescribed trajectory dynamically in response to the locally measured interaction force. The neural controllers adjust their weights independently using the RL algorithm of Gullapalli [9], a RL algorithm that has not been tested before in a multiagent learning environment. The learning goal is for each actuator to acquire, by maximizing the immediate reward, a cooperative intelligent positioning strategy that reduces the interaction force as much as possible while maintaining the desired trajectory. The undesirable internal

This work was supported by the Natural Sciences and Engineering Research Council (NSERC) of Canada. M. Karpenko and N. Sepehri are with the Department of Mechanical and Manufacturing Engineering, University of Manitoba, Winnipeg, Manitoba, CANADA R3T 5V6. J. Anderson is with the Department of Computer Science at the same University. e-mail: {karpenko@cc|andersj@cs|nariman@cc}.umanitoba.ca

1-4244-0210-7/06/$20.00 ©2006 IEEE

force is also reduced as a result.

II. Mathematical Modeling

A schematic of the servovalve controlled hydraulic actuator is shown in Fig. 1 along with the nomenclature associated with the mathematical model. State equations that describe the closed-loop actuator dynamics between the desired, x^d, and actual, x, positions of the actuator can be formed as [10], [11].

$$\begin{aligned}
\dot{x} &= v \\
\dot{v} &= \frac{1}{m}\left(-bv + A_1P_1 - A_2P_2 - F\right) \\
\dot{P}_1 &= \frac{\beta}{A_1x + \bar{V}}\left(Q_1 - A_1\dot{x}\right) \\
\dot{P}_2 &= \frac{\beta}{A_2(L-x) + \bar{V}}\left(-Q_2 + A_2\dot{x}\right) \\
\dot{x}_v &= v_v \\
\dot{v}_v &= -{\omega_v}^2x_v - 2\zeta_v\omega_v v_v + K_pk_v{\omega_v}^2(x^d - x)
\end{aligned} \tag{1}$$

Referring to (1), the system states are actuator position x, actuator velocity v, chamber pressures P_1 and P_2, valve spool displacement x_v, and valve spool velocity v_v. Parameters m and b are the combined mass of the actuator piston and rod and the viscous actuator damping, respectively. Area A_1 refers to the area of the piston and area A_2 is the annulus area of the piston on the rod side of the actuator. The effective bulk modulus of the hydraulic fluid is denoted by β, while L denotes the actuator stroke and parameter $\bar{V}$ represents the volume of the connecting lines between the servovalve and the actuator. The valve spool dynamics are expressed as a second-order lag where k_v is the valve spool position gain and parameters ω_v and ζ_v are the servovalve undamped natural frequency and damping ratio, respectively. Gain K_p is the closed-loop proportional gain. Load force F is a disturbance input and arises when the actuator interacts with the environment or another actuator.

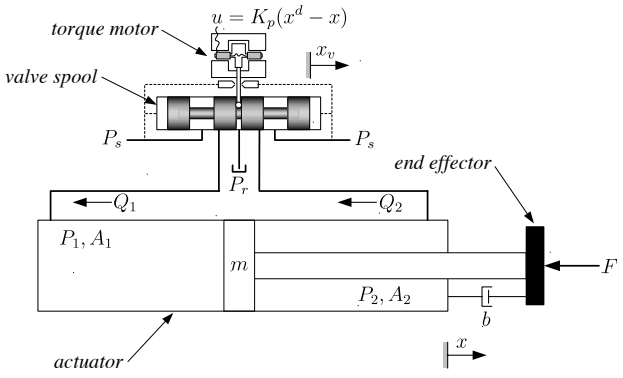

Fig. 1. Schematic of hydraulic actuator for mathematical modeling.

Assuming the valve orifices are matched and symmetrical, the control flows, Q_1 and Q_2, supplied by the servovalve are given by the turbulent orifice equation [10]. For positive valve spool displacements, $x_v \geq 0$, the control flows are

$$Q_1 = K_f x_v\sqrt{\frac{2\left(P_s - P_1\right)}{\rho}}; \quad Q_2 = K_f x_v\sqrt{\frac{2\left(P_2 - P_r\right)}{\rho}} \tag{2}$$

For negative valve spool displacements, $x_v < 0$, the control flows are

$$Q_1 = K_f x_v\sqrt{\frac{2\left(P_1 - P_r\right)}{\rho}}; \quad Q_2 = K_f x_v\sqrt{\frac{2\left(P_s - P_2\right)}{\rho}} \tag{3}$$

In (2) and (3), parameter ρ is the mass density of the hydraulic fluid and K_f is the servovalve flow gain. P_s and P_r denote the hydraulic supply and return pressures, respectively. The values of the hydraulic actuator parameters used for simulation are listed in Table I.

TABLE I

Parameters of hydraulic actuator for simulation.

Parameter	Symbol	Value
supply pressure	P_s	6.9 MPa (1000 psi)
return pressure	P_r	0
total mass of piston and rod	m	10 kg
viscous damping coefficient	b	350 N·sec/m
actuator stroke	L	0.6096 m
piston area	A_1	1.14×10^{-3} m^2
piston annulus area	A_2	6.33×10^{-4} m^2
volume of connecting lines	$\bar{V}$	4.15×10^{-5} m^3
valve flow gain	K_f	0.01245 m^2/m
valve spool position gain	k_v	4.06×10^{-5} m/V
valve natural frequency	ω_v	150 Hz
valve damping ratio	ζ_v	0.5
density of hydraulic fluid	ρ	847 kg/m^3
bulk modulus of hydraulic fluid	β	689 MPa
position control gain	K_p	250 V/m

The dynamic model of the manipulated object is constructed with reference to Fig. 2. Referring to Figs. 2a and 2b, the object of length, l is modeled as a lumped mass, M, connected to the end effector of each hydraulic actuator by a spring/damper pair having constants k and d, respectively. Since the hydraulic actuators cannot follow their desired trajectories perfectly, in general $x_1 \neq x_M^d - l/2$ and $x_2 \neq x_M^d + l/2$ leading to deformation of the object as in Fig. 2c. In this paper, the hydraulic actuators are each assumed to have the same values of the model parameters listed in Table I. However, the asymmetry in the actuator piston areas causes them to travel at different speeds depending on the direction of motion. This discrepancy causes differences in the position errors of each actuator and results in the deformation of the object. In practice, slight differences in the parameters of each actuator would further influence the position errors.

The equation of motion for mass M can be written as $M\ddot{x}_M = F_1 - F_2$, where F_1 and F_2 denote the interaction forces arising from the hydraulic actuators in contact with the object. Thus, the following state equations for simulation may be derived using the nomenclature of Fig. 2

$$\begin{aligned}
\dot{x}_M &= v_M \\
\dot{v}_M &= M^{-1}\left[k(x_1 + x_2 - 2x_M) + d(v_1 + v_2 - 2v_M)\right]
\end{aligned} \tag{4}$$

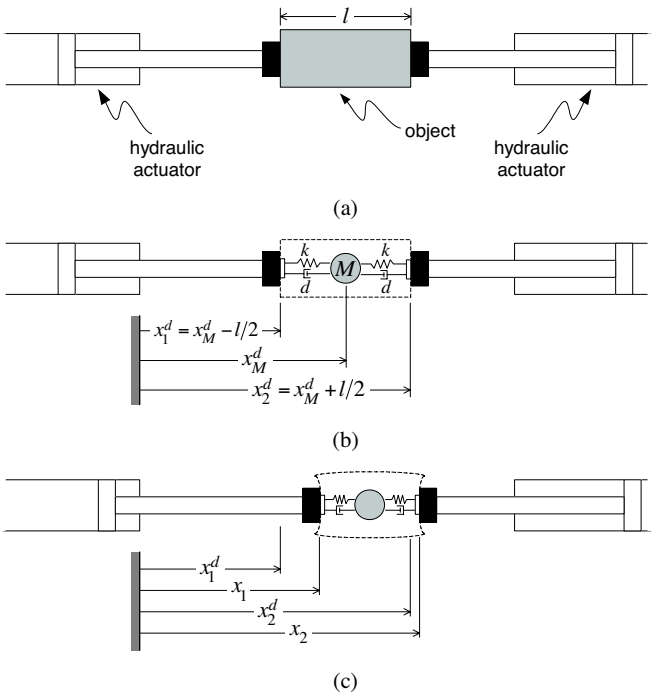

Fig. 2. Schematic of coordinated manipulation task for mathematical modeling: (a) hydraulic actuators grasping object of length l; (b) definition of desired trajectories x_1^d and x_2^d with respect to desired object position x_M^d; (c) deformation of object due to positioning errors.

In (4), v_M is the velocity of the mass while v_1 and v_2 refer to the velocities of the hydraulic actuators. The parameters of the object used for simulation are listed in Table II.

TABLE II

PARAMETERS OF OBJECT FOR SIMULATION.

Parameter	Value
M	0.296 kg
k	3002 N/m
d	47 N·s/m
l	120 mm

III. CONTROL ARCHITECTURE

A schematic of the reinforcement learning (RL) control architecture is illustrated in Fig. 3. With reference to Fig. 3, each actuator is positioned using a simple closed-loop proportional control law. The input signal to the control loop, $\bar{x}^d$, is the summation of the desired position trajectory, x^d, and a trajectory correction, Δx, that is output by the reinforcement learning neural network (RLNN). The trajectory correction is used to modulate the interaction force during the manipulation task. The RLNN consists of a temporal preprocessor, two independent multi-layer feedforward networks and a Gaussian random number generator. Output Δx is therefore a normally distributed random variable $\Delta x \sim \Psi(\mu, \sigma)$ whose mean value and stochasticity are controlled by neural network outputs, μ and $\sigma = s(\hat{r})$, respectively. Function $\sigma = s(\hat{r})$ was selected to be a monotonically decreasing nonnegative function

$$\sigma = 1 - \hat{r} \quad \text{(mm)} \tag{5}$$

Physically, equation (5) limits the stochastic exploration of the random output to an envelope within 2 mm of network output, μ , 95% of the time.

Referring to Fig. 3, it is observed that the single input to the RLNN is the measured force, F, between the end effector of the manipulator and the object. Temporal information carried by the force transducer output is implicitly captured by the temporal preprocessor which is a first-order tapped delay line consisting of unit delay operator z^{-1}. Hence, each neural network has two inputs, the current value of the force transducer output, $F(t)$, and the stored value of the force measurement, $F(t - \Delta t)$, from the previous time step. The short-term memory characteristic of the temporal preprocessor allows the neural network to generate an input-output map that considers the dynamics of the input and thus behave as a one-step ahead predictor [12]. Considering this and the fact that the neural networks have bias terms, the control action tends to resemble a proportional-integral-derivative control law.

The performance of each actuator is evaluated based on the measured interaction force, F, as follows:

$$r = \begin{cases} -0.05\,|F| + 1 & \text{if } |F| < 10 \text{ N} \\ -0.0075\,|F| + 0.575 & \text{if } 10 \text{ N} \le |F| < 50 \text{ N} \\ -0.004\,|F| + 0.4 & \text{if } 50 \text{ N} \le |F| < 100 \text{ N} \\ 0 & \text{otherwise} \end{cases} \tag{6}$$

Evaluation function (6) is piece-wise continuous and generates a reinforcement signal, $r \in [0, 1]$. Since it is assumed the each actuator is equipped with a force transducer, a reinforcement signal can be computed locally by substituting the measured value of force F_1 or F_2 for F in (6).

The multi-layer feedforward networks of each RLNN were restricted to have a variable number of hidden neurons arranged in a single layer. The hidden layer neurons use logistic activation functions $\phi(v) = \frac{1}{(1-e^{-v})}$. The output of the first neural network, referred to hereafter as the control network, is computed as the linear weighted sum of the outputs of the hidden layer neurons. The output unit of the second neural network, referred to in the sequel as the reinforcement estimator, uses a logistic activation function similar to the hidden layer neurons. This output function was selected since the actual value of the environmental reinforcement, r, varies continuously between 0 and 1. Together the output neurons of the control network and the reinforcement estimator as well as the Gaussian random number generator form a stochastic real valued (SRV) reinforcement learning element [9].

Both weight vectors of the SRV unit are tuned based on the actual value of the external reinforcement, $r \in [0, 1]$, received from the environment. In this paper, the hidden layer weights of the control network are updated at each time step, Δt, using a slightly modified version of the original SRV weight update rule [9]

$$w_j(t+\Delta t) = w_j(t) + 0.5\sigma(t)\mathcal{S}\left\{r(t) - \hat{r}(t)\right\}\left(\Delta x(t) - \mu(t)\right) z_j(t) \tag{7}$$

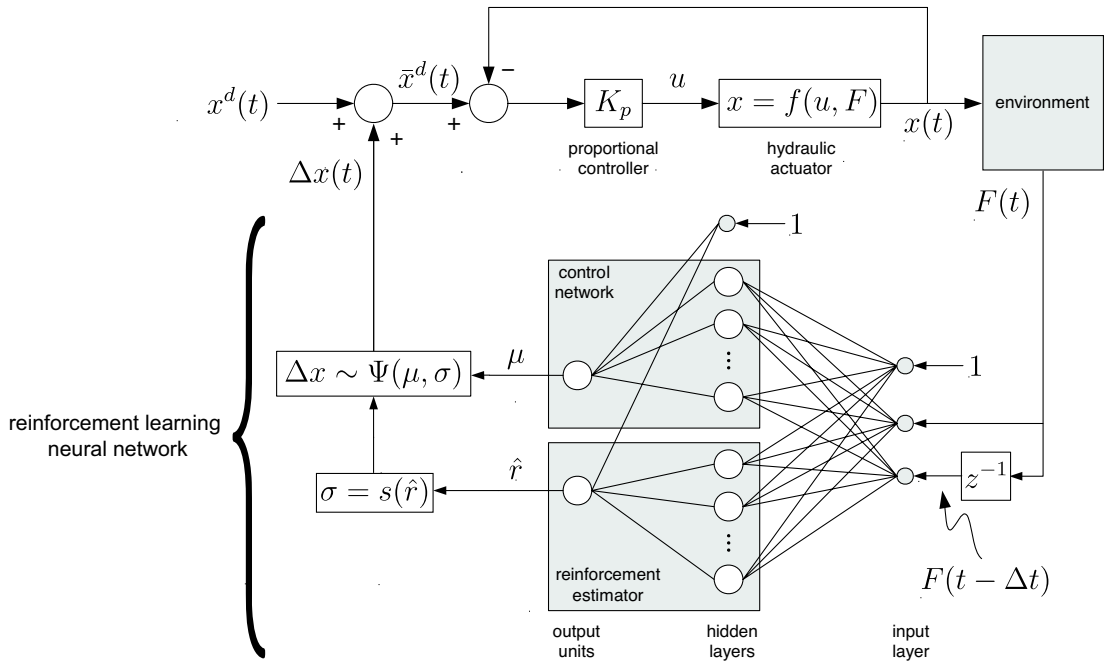

Fig. 3. Schematic of reinforcement learning control system.

where function $\mathcal{S}\{\cdot\}$ denotes the sign function. A more detailed discussion on the interpretation and comments on the stability of (7) can be found in [9] and [13], respectively.

Equation (7) was used only to update the hidden layer weights of the control network. The remaining input layer weights of the control network are adapted using standard error back-propagation [12] with a learning rate of 0.5. Since reinforcement comparison $r - \hat{r}$ is known at each time step, the weights of the reinforcement estimator were adapted using simple gradient descent and a learning rate of 0.5.

IV. Simulation Results

It is expected that the performance of the hydraulic actuators, with respect to alleviating the interaction force, should continue to improve as the manipulation task is repeated. Therefore, to facilitate training, the desired smooth path between the initial and final positions of the manipulated object was approximated by the continuous function

$$x_M^d = 200 \sin(0.5\pi t) + 300 \quad \text{(mm)} \tag{8}$$

The use of (8) allows the simulation of repetitive carrying tasks so that learning can proceed uninterrupted.

The first set of simulations performed were carried out to ascertain the ideal number of units in the hidden layers of each RLNN. The adjustable weights were first randomly initialized on the interval $[-0.5, 0.5]$ and the dynamic equations were simulated for 55 repetitions of the manipulation task (120 sec). The fourth-order Runge-Kutta integration scheme with a time step of 0.001 sec was used. Learning was assumed to occur at a constant rate of 100 Hz and an equal number of hidden neurons in the control network and reinforcement estimator was used. One hundred trials were run for hidden layer sizes ranging from 2 to 24 neurons. For each trial the root-mean-square (RMS) values of the interaction forces were computed as an indicator of their variance over the manipulation task. It was observed that networks having 10 hidden layer neurons were ideal for the reinforcement learning problem at hand.

To test the ability of the proposed reinforcement learning scheme to improve the coordination of the hydraulic actuators by reducing the interaction forces, a second set of simulations was performed with neural networks having the ideal hidden layer size. As before, the adjustable network weights were first initialized with random numbers on the interval $[-0.5, 0.5]$ and learning was assumed to occur at a rate of 100 Hz. The system was simulated for 120 sec and 500 trials were conducted in an effort to nullify the effects of the random initialization of the adjustable weights. For each trial, the average RMS value of the interaction force was computed over the last ten seconds of simulated time to assess performance after learning had converged. The results are shown in Fig. 4.

Fig. 4 shows that reinforcement learning improves the coordination of the hydraulic actuators and significantly reduces the RMS value of the interaction force (due to positioning errors) from a baseline value of 8.7 N (represented by the dashed line in Fig. 4) to 1.2 N on average. Note that the interaction force can never be zeroed entirely since some force is required to accelerate the object as the actuators follow the prescribed position trajectory. Moreover, the improvement in the coordination of the hydraulic manipulators has been accomplished using a decentralized control scheme and without the need to communicate local state information.

To illustrate how the performance of the hydraulic actuators is improved by reinforcement learning, the time histories

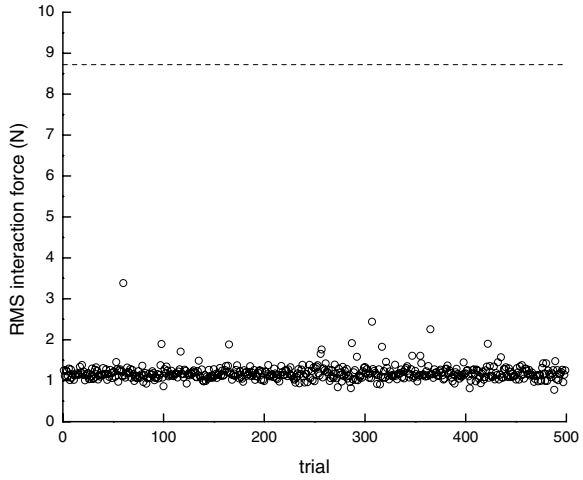

Fig. 4. Root-mean-square interaction force after learning. Dashed line indicates baseline performance with reinforcement learning neural networks switched off.

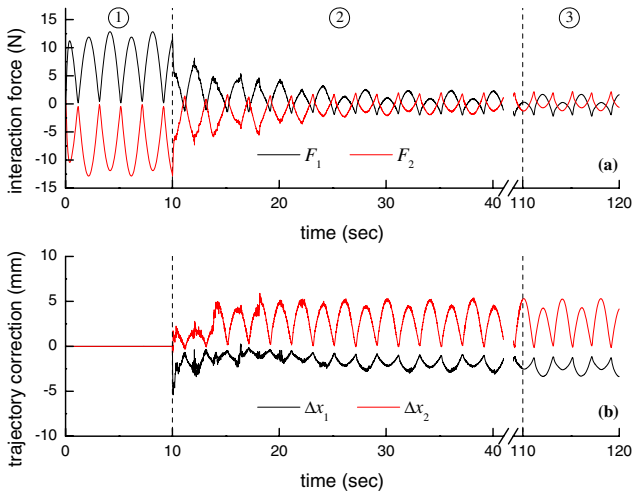

Fig. 5. Typical performance on manipulation task: (a) interaction forces, F_1 and F_2; (b) trajectory corrections, Δx_1 and Δx_2. Legend: ① before learning; ② during learning; ③ after learning.

of the interaction forces and the trajectory corrections applied by the RLNNs were obtained for a typical trial. The object internal force, which represents the excess interaction force absorbed by the object [1], and the object position error were also recorded. The RLNNs were switched off for the first ten seconds of the simulation to establish the baseline performance. Then, the RLNNs (with random initial conditions) were activated and learning was allowed for the next 100 seconds of simulated time. For the final ten seconds of the simulation, no further learning was allowed and the hydraulic actuators handled the object using their previously learned behaviors.

Fig. 5a shows that before learning, (time interval ①) the peak values of the interaction forces reached nearly 13 N in magnitude. After the RLNNs were switched on and learning began (time interval ②), the interaction forces were reduced almost immediately. Learning continued to improve the performance as time increased. After learning, during time interval ③, it is observed that the peak to peak amplitude of the interaction forces has been reduced to less than 3.5 N, a 70% improvement over the baseline performance.

The trajectory corrections applied by the RLNNs are shown in Fig. 5b. At the beginning of time interval ②, the RLNNs are exploring the environment most. Consequently, randomness in the corrections is observed. However, as the time increased beyond 30 seconds the noise associated with the explorative behavior is reduced. Thus, each RLNN has learned a control strategy that maximizes the reward.

Fig. 6 shows the internal force and error in the position $(x_M^d - x_M)$ of the object for the same trial. Inspection of the figure shows that the internal force is reduced from 12 N to less than 3 N without affecting the positional accuracy of the manipulated object. Thus, the benefit of improved coordination amongst the hydraulic actuators is obtained while adhering to the desired position trajectory.

A final simulation was carried out to test the generalization

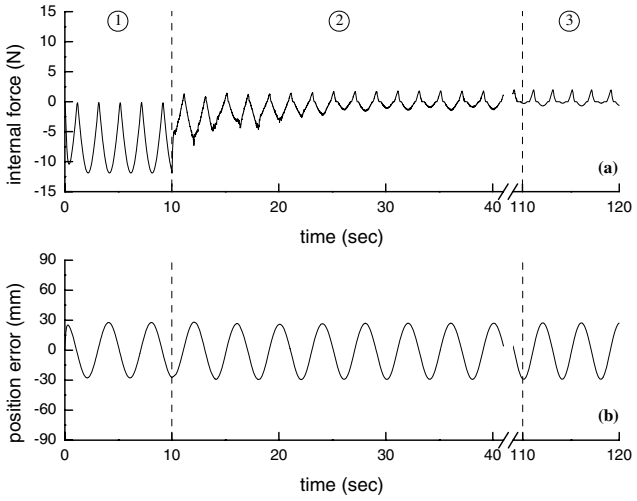

Fig. 6. Typical performance on manipulation task: (a) internal force; (b) position error $x_M^d - x_M$. Legend: ① before learning; ② during learning; ③ after learning.

of the previously learned control actions to position trajectories with starting and stopping motions before and after the manipulation task. The first 30 seconds of the repetitive test trajectory is shown in Fig. 7. To establish the baseline performance, the first three motions of Fig. 7 were carried out with the RLNNs turned off. Then at $t = 15$ seconds, the RLNNs, pre-trained using sinusoidal reference trajectory (8) for 30 seconds, were activated and no additional learning was allowed to occur. Finally, at $t = 30$ seconds learning was allowed to resume at a rate of 100 Hz. The results are shown in Fig. 8

Referring to Fig. 8 it is observed that after pre-training, the selected trajectory corrections generalize well to more realistic manipulation tasks and the hydraulic manipulators behave stably before starting and after stopping motion. Moreover, without further training (time interval ②), the

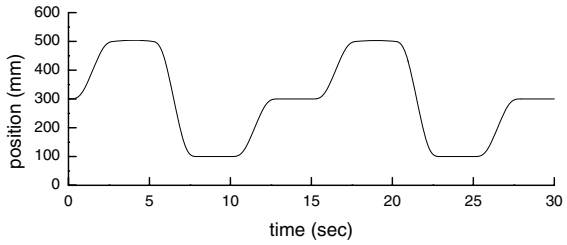

Fig. 7. Test trajectory, x_M^d, for trajectory following test.

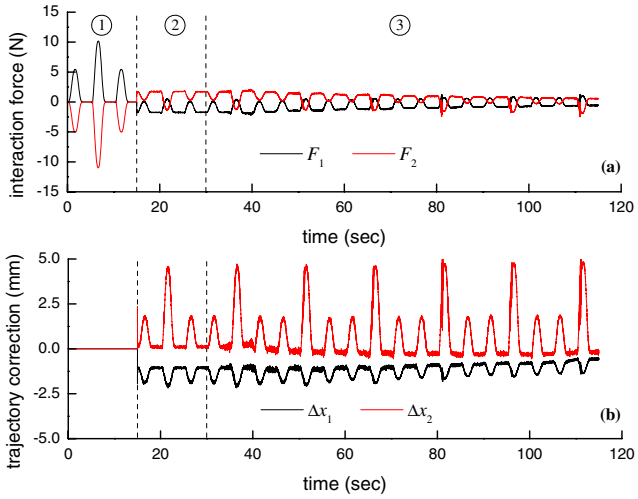

Fig. 8. Typical performance on trajectory following test: (a) interaction forces, F_1 and F_2; (b) trajectory corrections, Δx_1 and Δx_2. Legend: ① before learning; ② after pre-training on sinusoidal trajectory for 30 seconds; ③ additional learning.

peak interaction forces are reduced by over 60% compared to the case where the RLNNs are switched off. With additional learning (time interval ③) the neural network weights are further refined to reduce the loading of the object when the motion is stopped.

Several other simulation tests were carried out to examine the robustness of the proposed coordination scheme for objects having various characteristics. The adaptability of the distributed reinforcement learning scheme was also tested by examining how the system responds to sudden changes in the characteristics of the object. The system was observed to possess the capacity accommodate various objects and also to react stably to sudden changes in the object parameters. However, the results of these simulations have been omitted for the sake of brevity.

V. Conclusions

A method, based on reinforcement learning, was introduced in this paper to improve the coordination of two horizontal hydraulic actuators engaged in positioning an object along a line. Each manipulator system was outfitted with a specially designed reinforcement learning neural network (RLNN) controller to generate a modification to the prescribed formation constrained trajectory in response to the locally measured interaction force. The weights of each RLNN were updated on-line using a reinforcement learning scheme, described previously in the literature, yet never tested in a multiagent reinforcement learning environment. A detailed mathematical model of the multi-actuator system was developed to facilitate simulations that proved the efficacy of the proposed coordinated manipulation scheme. The simulation results showed that the manipulators could learn, by reinforcement, to intelligently select control actions that significantly reduce the interaction forces during the positioning of an object, while maintaining the desired position trajectory, even in unseen circumstances. Importantly, it was also observed that undesirable internal forces could be reduced using a decentralized control scheme and without the need for explicit communication of local state information.

References

[1] M. Vuckbrotovic and A. I. Tuneski, "Mathematical model of multiple manipulators: Cooperative compliant manipulation on dynamical environments," *Mechanism and Machine Theory*, vol. 33, pp. 1211–1239, 1998.

[2] B. M. Braun, G. P. Starr, J. E. Wood, and R. Lumia, "A framework for implementing cooperative motion on industrial controllers," *IEEE Transactions on Robotics and Automation*, vol. 20, pp. 583–589, 2004.

[3] R. Sutton and A. Barto, *Reinforcement Learning*. Cambridge, MA: MIT Press, 1998.

[4] C. W. Anderson, "Learning to control an inverted pendulum using neural networks," *IEEE Control Systems Magazine*, vol. 9, no. 3, pp. 31–37, 1989.

[5] V. Gullapalli, J. A. Franklin, and H. Benbrahim, "Acquiring robot skills via reinforcement learning," *IEEE Control Systems Magazine*, vol. 14, pp. 13–24, 1994.

[6] H. Sun and T.-C. Chiu, "Motion synchronization for dual-cylinder electrohydraulic lift systems," *IEEE/ASME Transactions on Mechatronics*, vol. 7, no. 2, pp. 171–181, 2002.

[7] T. Asokan, M. Singaperumal, and G. Seet, "Performance analysis of an electrohydraulic impedance controller for robotic interaction control," *Advanced Robotics*, vol. 17, pp. 791–806, 2003.

[8] H. Zeng and N. Sepehri, "Nonlinear position control of cooperative hydraulic manipulators handling unknown payloads," *International Journal of Control*, vol. 78, pp. 196–207, 2005.

[9] V. Gullapalli, "Stochastic reinforcement learning algorithm for learning real-valued functions," *Neural Networks*, vol. 3, no. 6, pp. 671–692, 1990.

[10] H. Merritt, *Hydraulic Control Systems*. New York: Wiley, 1967.

[11] M. Karpenko and N. Sepehri, "Robust position control of an electrohydraulic actuator with a faulty actuator piston seal," *ASME Journal of Dynamic Systems, Measurement and Control*, vol. 125, no. 3, pp. 413–423, 2003.

[12] C. M. Bishop, *Neural Networks for Pattern Recognition*. Oxford University Press, 1995.

[13] R. J. Williams, "Simple statistical gradient-following algorithms for connectionist reinforcement learning," *Machine Learning*, vol. 8, no. 3, pp. 229–256, 1992.