'90 韓國自動制御學術會議論文集 1990. 10.26~27

신경회로망을 사용한 로보트 매니퓰레이터의 학습 제어

° 경 계 현　　고 명 삼　　이 범 희

로보틱스 및 지능 시스템 연구실
서울대학교 대학원 제어계측공학과

Learning Control of a Robot Manipulator Using Neural Networks

Kye-Hyun Kyung　Myoung-Sam Ko　Bum-Hee Lee

Robotics and Intelligent Systems Laboratory
Dept. of Control and Instrumentation Engineering
Seoul National University

ABSTRACT : Learning control of a robot manipulator is proposed using the backpropagation neural network. The learning controller is composed of both a linear feedback controller and a neural network-based feedforward controller. The stability analysis of the learning controller is presented. Three energy functions are selected in teaching the neural network controller : $\frac{1}{2}\Sigma\{$ |torque error|^2 $\}$, $\frac{1}{2}\Sigma\{\ \alpha$|position error|^2 + β|velocity error|^2 $\}$, and $\frac{1}{2}\Sigma\{\ \alpha$|position error|^2 + β|velocity error|^2 + γ|acceleration error|^2 $\}$ and learning methods are presented. Simulation results show that the learning controller which is learned to minimize the third energy function performs better than the others in tracking problems. Some properties of the learning controller are discussed with simulation results.

1. 서 론

그동안 로보트 매니퓰레이터를 제어하기 위한 많은 방법들이 연구되어 왔다. 그중 한 방법은 매니퓰레이터의 관절 구동 토크를 계산하는데 로보트의 동력학 모델을 사용하는 방법이다(computed torque control methods) [1],[2]. 이 방법은 로보트의 동력학 모델이 완전하게 알려졌다면 아주 좋은 결과를 보이나, 동력학 모델 계수의 작은 오차에도 큰 제어 오차를 보이는 결점을 갖는다.

두번째로는 적응제어 기법을 로보트 제어 문제에 응용하는 방법들이 있다[3]-[6]. 이들 적응 제어 방식들은 일반적으로 로보트 동력학에 대한 사전 지식을 필요로 하지 않는 장점을 가지나, 실시간 파라미터 추정을 위한 계산량과 잡음에 대한 민감도가 시스템의 상태 변수가 증가함에 따라 증가하는 단점을 갖는다.

다음으로는 반복 학습 제어 방법이 있다[7],[8]. 이 방법은 로보트 동력학에 대한 지식을 필요로 하지 않으며 또한 안정성이 보장되는 장점을 지니나, 단지 반복적인 동작에만 적용할 수 있다는 단점을 갖는다.

한편 최근에 관심이 증대되고 있는 신경회로망을 로보트 제어에 이용하는 방법들이 제안되고 있다[9]-[12]. 신경회로망은 학습에 의해 주어진 과제를 수행하게 되므로 이 방법에 의한 로보트 제어는 로보트 동력학에 대한 사전 지식을 필요로 하지 않는다. Miyamoto 등[9]은 선형 되먹임 제어기와 그들이 제안한 신경회로망 앞먹임 제어기로 구성된 로보트 제어기를 제안하였는데 이 제어기에서 신경회로망의 학습을 위한 오차 신호는 되먹임제어기의 출력(되먹임 토크)으로 구성된다. 이 제어기에서 되먹임 변수는 로보트 각 관절의 위치와 속도가 된다. 이 제어기의 단점은 신경회로망의 입력을 계산하는데 로보트의 동력학식에 있는 부분식들을 사용하므로 신경회로망 입력의 계산량이 많은 것에 있다. Miller III 등[10],[11]은 CMAC(Cerebellar Model Articulation Controller)[13] 신경회로망을 사용한 로보트 제어 방법을 제안하였는데 이 모델에서 신경회로망은 로보트의 동력학 모델을 형성하도록 on-line으로 학습된다. 이 모델에서 신경회로망의 학습은 매니퓰레이터의 각 관절의 위치와 속도뿐만아니라 가속도까지 이용된다. Nam 등[12]은 역시 CMAC 신경회로망을 사용한 로보트 제어 방법을 제안하였는데 이 제어기는 Miyamoto등이 제안한 제어 구조와 같이 선형 되먹임 제어기와 신경회로망 앞먹임 제어기로 구성되며 되먹임 토크에 의해 학습이 수행된다. 이 모델에서 신경회로망의 학습에 가속도는 사용되지 않는다.

본 논문에서는 backpropagation 신경회로망[14]을 사용한 로보트 제어 방법에 관해 설명한다. 로보트 제어기는 선형 앞먹임 제어기와 신경회로망 되먹임 제어기로 구성되며 신경회로망 제어기의 학습은 첫째 Miller III 등[11]이 사용한 학습 방법, 둘째 Miyamoto 등[9]과 Nam 등[12]이 사용한 학습 방법, 그리고 마지막으로 본 논문에서 제안한 매니퓰레이터 각 관절의 위치와 속도 그리고 가속도 오차들을 이용한 학습 방법의 세 방법으로 수행된다. 2장에서는 신경회로망을 이용한 학습 제어기의 안정성과 수렴성을 로보트 매니퓰레이터의 동력학 모델과 함께 설명하고, 3장에서는 학습제어기 구조및 학습 방법을 설명한다. 4장에서는 시뮬레이션 결과를 예와 함께 설명한다. 5장에서는 결론을 제시한다.

2. 신경회로망을 사용한 로보트 매니퓰레이터의 학습제어기

일반적으로 n 자유도를 갖는 로보트 매니퓰레이터의 동력학은 (1)식으로 표현된다.

$$D(\theta)\ddot{\theta}(t) + h(\theta,\dot{\theta}) + c(\theta) = \tau(t) \qquad (1)$$

이 식에서 $\theta \in R^n$ 는 관절 좌표를 나타내며, $D(\theta) \in R^{n\times n}$ 는 항상 positive definite인 관성 행렬을, $h(\theta,\dot{\theta}) \in R^n$ 는 Coriolis와 centrifugal 힘과 관련이 있는 행렬을, 그리고 $c(\theta) \in R^n$는 중력 벡터를 나타낸다. $\tau(t) \in R^n$ 는 일반화된 힘 벡터를 나타낸다.

이 때 매니퓰레이터 관절의 목표 궤적의 상태를 $S_d(\theta_d, \dot{\theta}_d, \ddot{\theta}_d)$라고 하고 매니퓰레이터의 상태를 $S_a(\theta_a, \dot{\theta}_a, \ddot{\theta}_a)$라고 하면 식(1)은 목표 궤적에 대하여 다음과 같이 선형화될 수 있다.

$$R(t)\ddot{\varepsilon}(t) + E(t)\dot{\varepsilon}(t) + F(t)\varepsilon(t) + n(\ddot{\varepsilon},\dot{\varepsilon},\varepsilon,t) + O(t) = \tau(t) \qquad (2)$$

이 식에서 $\varepsilon(t)=\theta_o-\theta_d$ 이며, $n(\dddot{\varepsilon},\ddot{\varepsilon},\dot{\varepsilon},\varepsilon,t)$는 $\ddot{\varepsilon},\dot{\varepsilon},\varepsilon$의 고차항으로 무시될 수 있다. R(t), E(t), F(t), 그리고 O(t)는 다음과 같다.

$$R(t) = D(\theta_d),$$

$$E(t) = [\partial h(\theta,\dot{\theta})/\partial\dot{\theta}]_{(\theta_d,\dot{\theta}_d)}$$

$$F(t) = [\partial D(\theta)/\partial\theta]_{(\theta_d)}\ddot{\theta}_d + [\partial h(\theta,\dot{\theta})/\partial\theta]_{(\theta_d,\dot{\theta}_d)} + [\partial c(\theta)/\partial\theta]_{(\theta_d)}$$

$$O(t) = D(\ddot{\theta}_d)\theta_d + h(\dot{\theta}_d,\theta_d) + c(\theta_d).$$

제어 입력 $\tau(t)$를 (3)식과 같이 하고 (2)식에 대입하면 (2)식은 (4)식으로 다시 표현된다.

$$\tau(t) = K_p\, e(t) + K_v\, \dot{e}(t) + u(t) \qquad (3)$$

$$R(t)\,\ddot{e}(t) + (E(t) + K_v)\,\dot{e}(t) + (F(t) + K_p)\,e(t) = O(t) - u(t) \qquad (4)$$

where $e(t) = \theta_d(t) - \theta_o(t)$.

(3)식에서 K_p는 위치 되먹임 이득을, K_p는 속도 되먹임 이득을 나타내는 positive-definite 일정 행렬이고, u(t)는 선형 제어기의 오차를 보상해주기 위해 첨가되어야만하는 입력이다. (4)식에서 K_p와 K_v를 F(t)와 E(t)에 비해 충분히 크게 하여 F(t)와 E(t)가 무시될 수 있도록 하면 R(t)는 positive definite 행렬이므로 (4)식의 오차 방정식의 특성근(characteris- tic roots)이 음의 실수부를 갖도록 할 수 있다. 즉 K_p와 K_v를 충분히 크게 선택하면 오차 방정식은 안정화될 수 있다. 또한 첨가 입력 u(t)가 목표궤적의 함수로 구성되는 O(t)와 같도록 선택된다면 오차는 '0'으로 점근적으로 수렴하게 된다. 그러나 로보트 매니퓰레이터의 동력학을 완전하게 알 수는 없으므로 O(t)는 마찬가지로 정확하게 계산될 수 없다. 따라서 본 논문에서는 신경회로망으로 구성되는 제어기의 출력 u(t)를 학습에 의해 O(t)에 수렴하도록 한다. 신경회로망 제어기의 출력이 O(t)에 완전히 수렴하게 되면 오차가 0가 되므로 고정 이득 제어기는 활동하지 않게 된다.

3. 학습 제어기의 구조 및 학습 방법

본 논문에서 구성한 제어기의 구조는 그림1과 같다. 그림1의 제어기는 (4)식의 오차 방정식이 안정화 되도록 선택된 고정 이득 K_p와 K_v로 구성된 되먹임 제어기와 backpropagation 신경회로망으로 구성된 앞먹임 제어기로 구성된다. (5)식으로 표현될 수 있는 앞먹임 제어기의 출력 u(t)는 O(t)에 수렴되도록 학습된다.

$$u(kT) = f(S_{dk}, S_{ok}) \qquad (5)$$

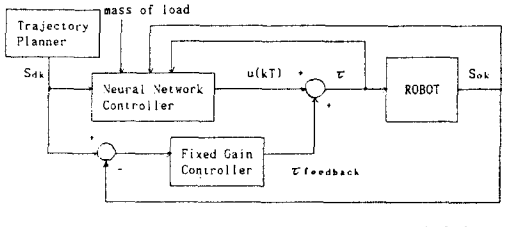


그림1 신경회로망을 이용한 로보트 매니퓰레이터 제어기 구성

식(5)에서 T는 샘플링 시간을 의미하며, S_{dk}와 S_{ok}는 k번째 샘플링 순간에서의 궤적 계획기에서 발생된 목표 궤적 상태와 매니퓰레이터의 되먹임 상태를 각각 나타낸다. 앞먹임 제어기의 학습은 세가지 방법으로 각각 수행되는데 그 방법들은 다음과 같다.

(1) 방법1 : 첫번째 학습 방법은 Miller III 등[10],[12]에 의해 제안된 방법이다. k번째 샘플링 순간의 시작에서 궤적 계획기에서 발생된 목표궤적 S_{dk}와 매니퓰레이터의 되먹임 상태(states) S_{ok}를 입력받는 로보트 제어기는 (3)식의 토크 입력 τ를 매니퓰레이터에 전달하고 매니퓰레이터는 입력 토크에 의해 S_{ok+1}로 이동하게 된다. 신경회로망 제어기의 학습은 k번째 샘플링 순간의 끝에서 이루어지는데, 먼저 $\tau' = f(S_{ok+1}, S_{ok})$가 계산되면, 신경회로망 제어기는 토크 오차 $(\tau-\tau')$에 의해 backpropagation으로 학습되게 된다. 즉 이 방법에 의해 학습되는 신경회로망 제어기는 매니퓰레이터가 S_{ok} 상태에 있고 목표궤적의 상태가 S_{dk} 라면 매니퓰레이터를 S_{ok} 상태로부터 S_{dk} 상태로 이동시키는 토크를 출력하도록 학습된다.

(2) 방법2 : 두번째 학습 방법은 Miyamoto 등[9]과 Nam[11]등에 의해 제안된 방법이다. k번째 샘플링 순간의 시작에서 (3)식의 토크 입력 τ가 매니퓰레이터에 전달되면 매니퓰레이터는 S_{ok} 상태로부터 S_{ok+1} 상태로 이동하게 된다. 그러면 k번째 샘플링 순간의 끝에서 $K_v(\dot{\theta}_{dk}-\theta_{ok+1}) + K_p(\dot{\theta}_{dk}-\theta_{ok+1})$가 계산되고 이 값을 backpropagation 시킴에 의해 신경회로망 제어기의 학습이 이루어진다. 즉 신경회로망은 고정 이득 되먹임 제어기의 출력을 극소화하도록 학습된다. 그런데 고정 이득 되먹임 제어기의 출력은 이득 상수에 곱해진 매니퓰레이터 관절의 위치 오차와 속도 오차의 합으로 구성되므로 이 방법에 의해 신경회로망은 위치 오차와 속도 오차를 극소화하도록 학습된다. 이 방법에서 가속도 궤적은 사용되지 않는다.

(3) 방법3 : 신경회로망 제어기의 세번째 학습 방법은 다음과 같다. 먼저 k번째 샘플링 순간의 시작에서 (3)식의 토크 입력 τ가 매니퓰레이터에 전달되면 매니퓰레이터는 S_{ok} 상태로부터 S_{ok+1} 상태로 이동하게 된다. 그러면 k번째 샘플링 순간의 끝에서 $\alpha(\theta_{dk}-\theta_{ok+1}) + \beta(\dot{\theta}_{dk}-\dot{\theta}_{ok+1}) + \gamma(\ddot{\theta}_{dk}-\ddot{\theta}_{ok+1})$가 오차로서 계산되고 backpropagation 학습 방법에 의해 신경회로망 제어기의 학습이 이루어진다. 즉 신경회로망 제어기는 위치 오차와 속도 오차 그리고 가속도 오차의 가중치화된 합을 극소화 하도록 학습된다. 이 방법에서 관측 가속도는 신경회로망의 학습에만 사용된다.

표1. 신경회로망 제어기의 학습 모델 비교
(m : 부하의 무게)

model	inputs to NNC	#FC	#BC	energe function
1	$\theta_d, \dot{\theta}_d, \ddot{\theta}_d, m$	2	1	$\frac{1}{2}\Sigma(\tau_d-\tau_o)^2$
2	$\theta_d, \dot{\theta}_d, \ddot{\theta}_d, \theta_o, \dot{\theta}_o, \tau_{o-1}, m$	2	1	$\frac{1}{2}\Sigma(\tau_d-\tau_o)^2$
3	$\theta_d, \dot{\theta}_d, \theta_o, \dot{\theta}_o, m$	2	1	$\frac{1}{2}\Sigma(\tau_d-\tau_o)^2$
4	$\theta_d, \dot{\theta}_d, \ddot{\theta}_d, \theta_o, \dot{\theta}_o, \tau_{o-1}, m$	1	1	$\frac{1}{2}\Sigma\{\alpha(\theta_d-\theta_o)^2 + \beta(\dot{\theta}_d-\dot{\theta}_o)^2\}$
5	$\theta_d, \dot{\theta}_d, \ddot{\theta}_d, \theta_o, \dot{\theta}_o, \tau_{o-1}, m$	1	1	$\frac{1}{2}\Sigma\{\alpha(\theta_d-\theta_o)^2 + \beta(\dot{\theta}_d-\dot{\theta}_o)^2 + \gamma(\ddot{\theta}_d-\ddot{\theta}_o)^2\}$
6	$\theta_d, \dot{\theta}_d, \ddot{\theta}_d, m$	1	1	$\frac{1}{2}\Sigma\{\alpha(\theta_d-\theta_o)^2 + \beta(\dot{\theta}_d-\dot{\theta}_o)^2 + \gamma(\ddot{\theta}_d-\ddot{\theta}_o)^2\}$

#FC : NUMBER OF FORWARD COMPUTATIONS PER EACH SAMPLING TIME
#BC : NUMBER OF BACKWARD COMPUTATIONS PER EACH SAMPLINK TIME
NNC : NEURAL NETWORK CONTROLLER

신경회로망 제어기의 학습은 위의 각 학습 방법에 대해서도 신경회로망의 입력을 어떻게 구성하는가에 따라 달라질 수 있다. 표1에는 본 논문에서 제안한 여섯가지 학습 모델을 제시한다. 표1에서 m은 부하의 무게를 나타낸다. 모델 1과 모델 2 그리고 모델 3은 방법1에 속하는데 매니퓰레이터 상태를 신경회로망 제어기의 입력에 포함시킬 것인가와 가속도 항을 포함시킬 것인가에 따라 구분된다. 모델 4는 방법2에 속한다. 모델 5와 모델 6은 방법3에 속하는데 매니퓰레이터 상태를 신경회로망 제어기의 입력에 포함시킬 것인가에 따라 그 구분이 이루어진다. 표1에서 energy function은 각 방법에 대해 backpropagation 학습 법칙에서 요구되는 목적 함수를 의미한다.

4. 시뮬레이션 결과

신경회로망을 이용한 학습 제어기의 성능을 평가하기 위해 그림2의 2 자유도를 갖는 로보트 매니퓰레이터의 제어에 학습 제어기를 적용하였다. 그림2의 매니퓰레이터의 동력학은 식(6)으로 표현된다.

$$\begin{bmatrix}\tau_1\\ \tau_1\end{bmatrix}=\begin{bmatrix}\frac{1}{3}m_1l_1^2+4/3m_2l_2^2+m_2C_2l_1l_2 & \frac{1}{3}m_2l_2^2+\frac{1}{2}m_2l_1l_2C_2\\ \frac{1}{3}m_2l_2^2+\frac{1}{2}m_2C_2l_1l_2 & \frac{1}{3}m_2l_2^2\end{bmatrix}\begin{bmatrix}\ddot{\theta}_1\\ \ddot{\theta}_1\end{bmatrix}$$
$$+\begin{bmatrix}-\frac{1}{2}m_2S_2l_1l_2\dot{\theta}_2^2-m_2S_2l_1l_2\dot{\theta}_1\dot{\theta}_2\\ \frac{1}{2}m_2S_2l_1l_2\dot{\theta}_1^2\end{bmatrix}$$
$$+\begin{bmatrix}\frac{1}{2}m_1gl_1C_1+\frac{1}{2}m_2gl_2C_{12}+m_2gl_1C_1\\ \frac{1}{2}m_2gl_2C_{12}\end{bmatrix} \quad (6)$$

식(6)에서 $C_1 \equiv \cos(\theta_1)$, $C_2 \equiv \cos(\theta_2)$, $S_2 \equiv \sin(\theta_2)$, 그리고 $C_{12} \equiv \cos(\theta_1+\theta_2)$ 으로 정의된다. 또한 m_i와 l_i(i=1,2)는 i번째 링크의 무게와 길이를 나타내며 g는 중력 상수를 나타낸다. 시뮬레이션에서 사용한 조건들은 다음과 같다.

sampling time: T = 5 (ms);
mass of link: m_1 = 10 (Kg); m_2 = 6 (Kg);
length of link: l_1 = 1 (m); l_2 = 1 (m);
feedback gain: K_P=diag(150,150) (N/rad);
K_V=diag(100,100) (Ns/rad);
desired trajectory: $\theta_1 = 5\pi/12 \cos(\pi t/3)$ (rad);
$\theta_2 = -\pi/3 \cos(\pi t/3) + 2\pi/5$ (rad);
initial condition: $\theta_1(0) = 5\pi/12$ (rad);
$\theta_2(0) = -\pi/3 + 2\pi/5$ (rad);
$\dot{\theta}_1(0) = \dot{\theta}_2(0) = 0$ (rad/sec);
momentum in backpropagation learninging rule: 0.9 .

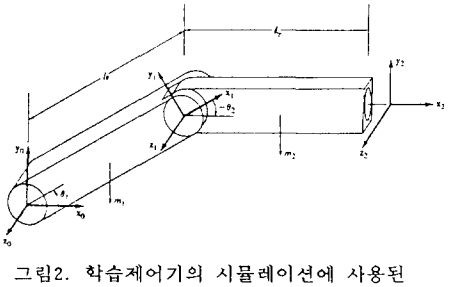


그림2. 학습제어기의 시뮬레이션에 사용된 로보트 매니퓰레이터

표2, 표3, 표4는 2장의 표1에서 설명한 여섯가지 모델에 대한 시뮬레이션 결과를 나타낸다. 이들 표에서 오차항은 식(7)과 같이 목표 궤적을 매니퓰레이터가 이동하는 한 주기 동안의 각 샘플링 시간에 구한 오차의 제곱값을 그 주기

표2. 학습 모델에 따른 학습 제어기의 매니퓰레이터 제어 성능 비교(η = 0.01 인 경우)

η = 0.01							
model	load (Kg)	#n	err_j1	err_j2	err_v1	err_v2	err_a1	err_a2
1	0.	2	1.475668	0.019289	4.829937	0.080203	32.998835	3.239779
	4.	2	4.845129	0.109059	26.684835	1.075109	285.375130	21.515548
2	0.	2	1.474254	0.019192	4.821354	0.079690	32.914365	3.233255
	4.	2	4.829622	0.108300	26.576337	1.068003	284.247571	21.531883
3	0.	2	1.469508	0.019192	4.805123	0.079686	32.816130	3.297189
	4.	2	4.817065	0.108323	26.568247	1.072328	285.331749	21.699402
4	0.	4	0.340415	0.032491	1.173693	0.062318	8.399764	0.758613
	4.	3	1.051945	0.104711	4.653583	0.412819	59.311573	14.298839
5	0.	2	0.035395	0.003607	0.127897	0.005956	0.744820	0.080366
	4.	2	0.102686	0.009837	0.397385	0.018094	2.707574	0.130877
6	0.	2	0.035052	0.003623	0.128135	0.005997	0.746374	0.050624
	4.	2	0.103021	0.009894	0.398904	0.018223	2.720067	0.131992

#n : NUMBER OF LEARNING ITERATIONS UNTIL CONVERGE

표3. 학습 모델에 따른 학습 제어기의 매니퓰레이터 제어 성능 비교(η = 0.03 인 경우)

η = 0.03							
model	load (Kg)	#n	err_j1	err_j2	err_v1	err_v2	err_a1	err_a2
1	0.	6	0.451280	0.009611	1.313445	0.026532	13.664268	2.838074
	4.	18	1.525673	0.020625	7.301153	0.227059	122.512458	9.831456
2	0.	5	0.348160	0.009752	0.980051	0.021984	8.506543	2.623029
	4.	17	1.521475	0.020396	7.199122	0.221001	119.209086	9.569855
3	0.	5	0.450391	0.009621	1.311822	0.026550	13.728642	2.941248
	4.	16	1.507587	0.020306	7.175276	0.224245	121.317245	9.790012
4	0.	4	0.264213	0.026221	0.928336	0.050650	5.513491	0.481361
	4.	3	0.781529	0.081353	4.217618	0.515603	130.374668	14.669728
5	0.	2	0.003837	0.000494	0.014123	0.000701	0.088743	0.011884
	4.	2	0.010851	0.001197	0.042533	0.001905	0.285739	0.019405
6	0.	2	0.003855	0.000497	0.014193	0.000708	0.089155	0.011924
	4.	2	0.010918	0.001206	0.042813	0.001926	0.287742	0.019545

#n : NUMBER OF LEARNING ITERATIONS UNTIL CONVERGE

표4. 학습 모델에 따른 학습 제어기의 매니퓰레이터 제어 성능 비교(η = 0.05 인 경우)

η = 0.05								
model	load (Kg)	#n	err_j1	err_j2	err_v1	err_v2	err_a1	err_a2
1	0.	3	0.450115	0.009659	1.275453	0.024534	8.945291	2.612570
	4.	4	3.890369	0.084841	21.053772	0.824067	258.228492	21.958386
2	0.	3	0.350437	0.009650	1.108712	0.023453	9.711155	2.754087
	4.	3	3.596007	0.078573	19.510602	0.800000	240.041855	20.201873
3	0.	3	0.490382	0.009695	1.396753	0.025972	10.371427	2.646243
	4.	11	1.497640	0.020018	6.749647	0.192963	103.880778	8.354925
4	0.	14	0.034148	0.003489	0.127928	0.024843	12.247407	6.041117
	4.	**						
5	0.	2	0.001347	0.000220	0.005103	0.000279	0.037536	0.008805
	4.	2	0.003820	0.000490	0.015234	0.000715	0.106941	0.011217
6	0.	2	0.001373	0.000224	0.005200	0.000285	0.038091	0.008830
	4.	2	0.003866	0.000496	0.015420	0.000726	0.108196	0.011293

#n : NUMBER OF LEARNING ITERATIONS UNTIL CONVERGE
** : NOT CONVERGE UNTIL 100TH LEARNING ITERATION

에 대해 모두 더한 값으로 하였고, 4Kg의 부하가 있는 경우와 무부하의 경우의 두가지 경우에 대해 오차항을 구하였다. 이 때 부하의 무게는 미리 안다고 가정하였다. 한편 이들 표에서 η는 backpropagation학습 법칙에서 사용되는 학습율(learning ratio)을 나타낸다.

$$err\_ji = \Sigma(\theta_{id}-\theta_{io})^2 \quad (7\text{-a})$$

$$err\_vi = \Sigma(\dot{\theta}_{id}-\dot{\theta}_{io})^2 \quad (7\text{-b})$$

$$err\_ai = \Sigma(\ddot{\theta}_{id}-\ddot{\theta}_{io})^2 \quad (7\text{-c})$$

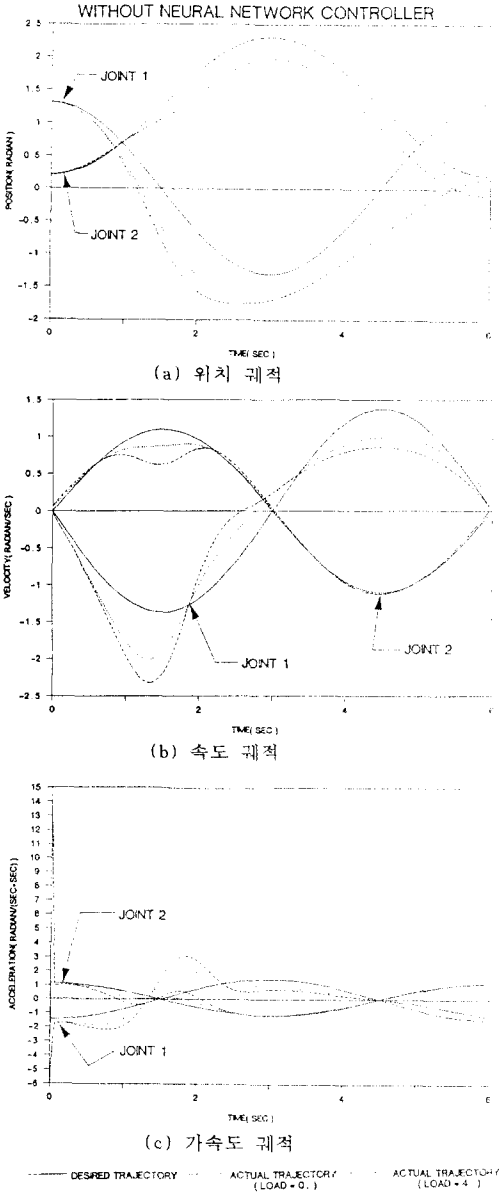
이들 표로부터 같은 모델에 대해서도 η에 따라 제어 효과에 큰 차이가 있음을 볼 수 있다. 또한 신경회로망 제어기의 입력의 차이에도 불구하고 같은 방법(2장에서 설명한 세가지 방법)에 의해 학습된 제어기는 비슷한 성능을 보임을 알 수 있다. 본 시뮬레이션으로부터 세번째 학습 방법(모델5와 모델6)으로 학습된 신경회로망 제어기가 가장 좋은 결과를 보임을 알 수 있다. 세번째 학습 방법에서 주의해야할 사항은 일반적으로 가속도가 위치나 속도항의 작은 오차에 대해서도 크게 변동하므로 γ를 α와 β에 비해 상대적으로 아주 작게 선택해야 한다는 점이다. 본 시뮬레이션에서 α, β, 그리고 γ의 값은 각각 1.5, 1.0, 0.1로 하였다. α와 β는 그대로 두고 γ를 0.5 또는 1.0으로 하였을 때 신경회로망 제어기는 수렴하지 못하고 발산하는 현상을 시뮬레이션을 통해 볼 수 있었다.

표5는 모델5의 학습 제어기의 α, β, 그리고 γ의 값에 따른 제어 성능의 비교를 나타낸다. 표5로부터 α=4.0, β=8/3, γ=0.1로 하였을 때 제안된 신경회로망 제어기는 가장 좋은 결과를 보임을 볼 수 있다.

그림3은 선형 되먹임 제어기만을 사용했을 때 매니퓰레이터의 실제 관절 궤적을 나타낸 것이다. 그림4는 모델5(α=1.5, β=1.0, γ=0.1)의 신경회로망 제어기를 추가했을 때의 실제 관절 궤적과 추적 오차를 나타낸다. 그림4에서 시간 0는 학습의 시작점을 나타낸다. 그림4로부터 학습제어기는 학습 시작 후 약 0.3초가 경과한 뒤부터 가속도 궤적까지 목표 궤적을 잘 추적함을 볼 수 있다. 그림4에서 부하가 4Kg일 때 최대 위치 오차는 약 0.003 rad(= 0.172 degree) 임을 알 수 있는데, 이것은 매니퓰레이터의 각 관

표5. 모델5의 에너지 함수에서의 가중치에 따른 학습 제어기의 성능 비교

MODEL 5 (η = 0.05, β = 0.9, second iteration)								
α	β	γ	load (Kg)	err_j1	err_j2	err_v1	err_v2	err_a1	err_a2
1.5	1.0	0.1	0.	0.001347	0.000220	0.005103	0.000279	0.037536	0.008805
			4.	0.003820	0.000490	0.015234	0.000715	0.106941	0.011217
2.0	4/3	0.1	0.	0.000733	0.000122	0.0027713	0.000153	0.024360	0.007671
			4.	0.002078	0.000270	0.008281	0.000390	0.062465	0.008971
3.0	2.0	0.1	0.	0.000314	0.000053	0.001185	0.000066	0.015326	0.006696
			4.	0.000890	0.000118	0.003538	0.000167	0.031688	0.007311
4.0	8/3	0.1	0.	0.000174	0.000030	0.000654	0.000037	0.012363	0.006305
			4.	0.000494	0.000068	0.001961	0.000093	0.022327	0.006802
5.0	10/3	0.1	0.	0.001709	0.000061	0.377043	0.037694	1206.341	108.5329
			4.	0.033059	0.003727	2.041046	0.288074	713.1151	97.45113
6.0	4.0	0.1	0.	5285.679	0.006734	1325.852	1.206643	7996.958	17073.96
			4.	3754.178	24.39588	5282.785	215.7407	39822.46	46277.09
4.0	8/3	0.08	0.	0.000422	0.000043	0.091113	0.009245	318.0412	27.44841
			4.	0.332092	0.003005	0.245483	0.026116	36.60752	6.853535
4.0	8/3	0.11	0.	0.000174	0.000031	0.000659	0.000038	0.012362	0.006481
			4.	0.000493	0.000068	0.001964	0.000095	0.021563	0.006731



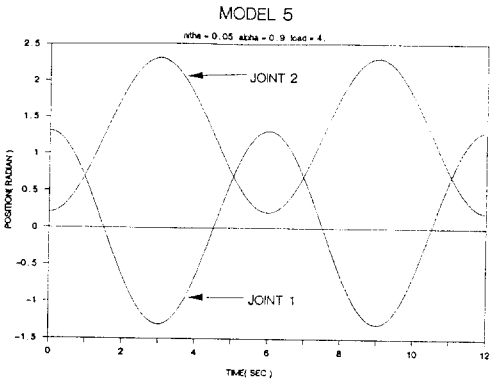
(a) 위치 궤적

(b) 속도 궤적

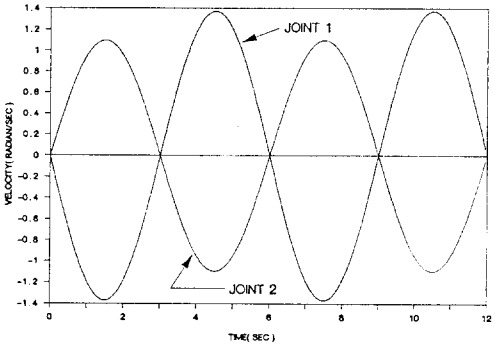
(c) 가속도 궤적

그림3. 선형 제어기만을 사용한 매니퓰레이터 제어

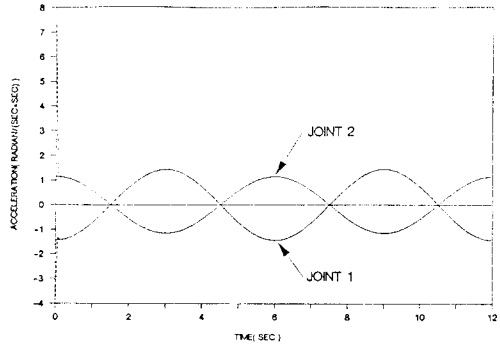
절의 기어비를 100:1이라고 가정하면 실제 매니퓰레이터의 각 관절의 위치 오차는 약 0.00003 rad(= 0.00172 degree) 이내가 됨을 의미한다. 즉 2자유도 매니퓰레이터의 두 관절의 길이를 각각 1m라고 했을 때 첫번째 관절의 오차에 의한 Cartesian 좌표계에서의 매니퓰레이터 손끝의 최대 오차는 약0.06mm가 된다. 그림5는 모델5(α=1.5, β=1.0, γ=0.1)의 경우 학습에 따른 신경회로망 제어기의 앞먹임 토크와 고정 이득 제어기로 구성된 선형 제어기의 되먹임 토크의 비교를 나타낸다. 그림5로부터 학습이 진행됨에 따라 위치와 속도 궤적이 목표 궤적에 수렴하게 되므로 되먹임 토크는 앞먹임 토크에 비해 상대적으로 아주 작게됨을 볼 수 있다. 즉 학습이 진행됨에 따라 신경회로망 제어기는 목표 궤적을 추적하기 위해 2장에서 요구되던 역할을 수행한다.



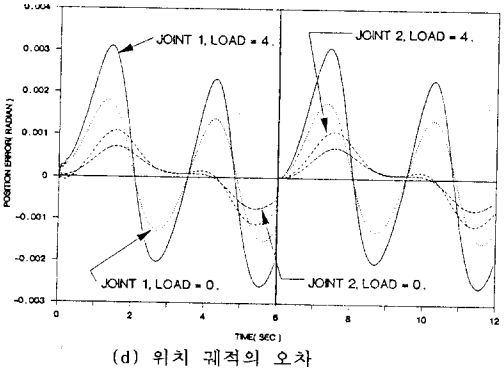
(a) 위치 궤적 (부하: 4Kg)
——— : DESIRED TRAJECTORY ····· : ACTUAL TRAJECTORY



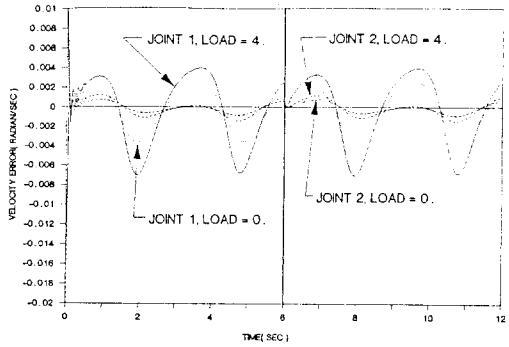
(b) 속도 궤적 (부하: 4Kg)
——— : DESIRED TRAJECTORY ····· : ACTUAL TRAJECTORY



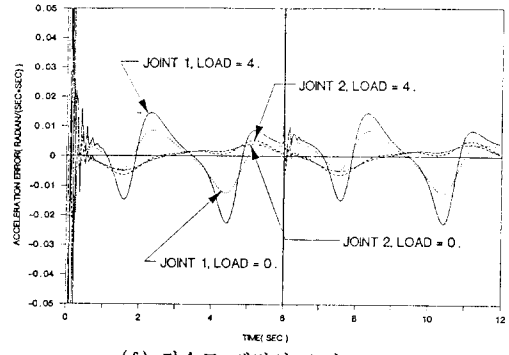
(c) 가속도 궤적 (부하: 4Kg)
——— : DESIRED TRAJECTORY ····· : ACTUAL TRAJECTORY



(d) 위치 궤적의 오차



(e) 속도 궤적의 오차



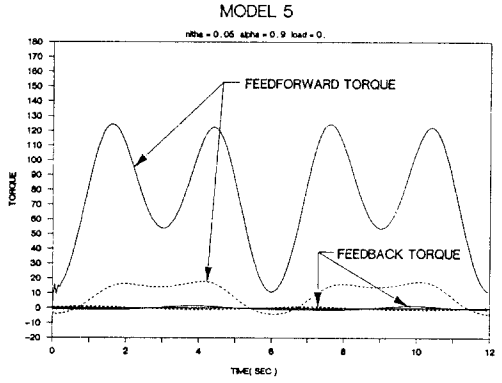
(f) 가속도 궤적의 오차

그림4. 제안된 학습제어기에 의한 매니퓰레이터 제어

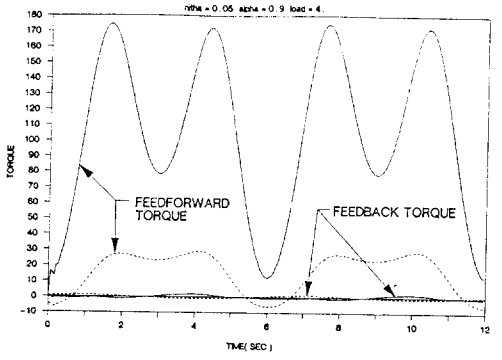
그림6은 모델5(α=1.5, β=1.0, γ=0.1)를 사용한 로보트 매니퓰레이터의 pick-and-place 작업 시뮬레이션이다. 시뮬레이션에서 신경회로망 제어기는 작업을 수행하기 전에 앞에서 서술한 목표궤적에 위해 단지 6초간 사전 학습을 받게 하였으며, 따라서 본 시뮬레이션에서 매니퓰레이터는 학습된적이 없는 궤적을 추적하게 된다. 이때 부하는 4.0 Kg으로 하였으며 관절 궤적은 sin함수를 이용 발생되도록 하였다. 시뮬레이션에서 제안된 학습제어기에 의해 학습되지 않은 궤적에 대해서도 또한 부하의 변동에 대해서도 거이 무관하게 매니퓰레이터를 작은 오차 한도 내에서 제어할 수 있음을 볼 수 있다.

5. 결 론

본 논문에서는 신경회로망을 이용한 로보트 매니퓰레이터의 학습 제어에 대해 서술하였다. 학습 제어기는 고정 이득 제어기로 구성된 선형 되먹임 제어기와 신경회로망 되먹임 제어기로 구성되는데, 수학적 분석을 통해 선형제어기는 시스템을 안정화하는 역할을 수행하며 신경회로망 제어기는 오차를 0에 수렴하도록 하는 역할을 수행함을 보였고 시뮬레이션을 통해 그것을 확인하였다. 본 논문에서 신경회로망 제어기는 backpropagation 신경회로망으로 구성하였다. 신경회로망 제어기의 학습은 첫째 Miller III 등[11]이 사용한 학습 방법, 둘째 Miyamoto 등[9]과 Nam 등[12]이 사용한 학습 방법, 그리고 세째 본 논문에서 제안한 매니퓰레이터 각 관절의 위치와 속도 그리고 가속도 오차들을 이용한 학습 방법의 세 방법으로 수행하여 본 논문에서 제안한 방법으로 가장 좋은 결과를 얻을 수 있음을 시뮬레이션



(a) 무부하인 경우

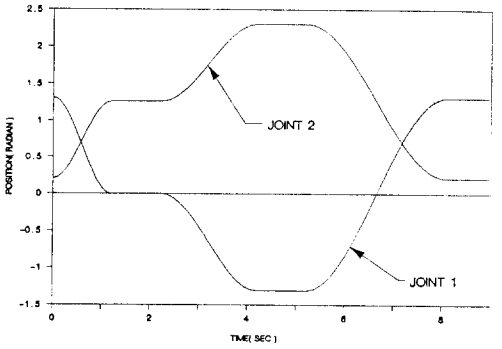


(b) 부하 = 4Kg 인 경우

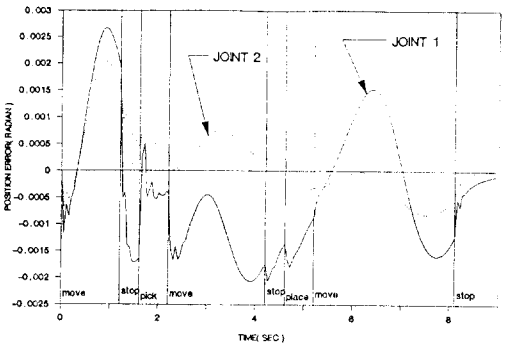
JOINT 1 ····· JOINT 2

그림5. 제안된 학습제어기를 사용한 매니퓰레이터 제어에서 신경회로망 제어기에 의해 출력된 토크와 선형 되먹임 제어기에 의해 출력된 토크의 비교

을 통해 보였다. 2자유도를 갖는 매니퓰레이터에 대한 시뮬레이션을 통해 제안된 학습 방법에 의해 학습제어기가 매니퓰레이터 각 관절의 위치 및 속도 궤적은 물론 가속도 궤적까지 목표 궤적을 잘 추적하도록 학습됨을 보였다.



(a) 위치 궤적 추적



(b) 위치 궤적 오차

그림6. 제안된 학습제어기를 사용한 로보트 매니퓰레이터 pick-and-place 작업 제어

참고 문헌

1. R. P. Paul, " Modelling, trajectory calculation and serving of a computer controlled arm, " Stanford Artificial Intelligence Lab. Memo AM-177, Nov. 1972.
2. J. Y. S. Luh, M. W. Walker, and R. P. Paul, "Resolved acceleration control of mechanical manipulator," IEEE Trans. Automat. Contr., vol. AC-25, pp. 468-474, 1980.
3. S. Dubowsky and D. T. DesForges, "The application of model referenced adaptive control to robot manipulators,"Trans. ASME, J. Dyn. Syst., Meas. Contr., vol. 101, pp. 193-200, 1979.
4. A. J. Koivo and T. H. Guo, " Adaptive linear controller for robotic manipulators," IEEE Trans. Automat. Contr., vol. AC-28, pp. 162-171, 1983.
5. C. S. G. Lee and M. J. Chung, "An adaptive control strategy for mechanical manipulators," IEEE Trans. Automat. Contr., vol. AC-29, pp. 837-840, 1984.
6. C. S. G. Lee and B. H. Lee, " Resolved motion adaptive control for mechanical manipulators, " Trans. ASME, J. Dyn. Syst., MEAS. Contr., vol. 106, no. 2, pp. 134-142, 1984
7. S. Arimoto, S. Kawamura, and F. Miyazaki,"Bettering operation of robots by learning, " J. Robotics Syst., vol. 1, pp. 123-140, 1984.
8. S. Kawamura, F. Miyazaki, and S. Arimoto, " Realization of robot motion based on a learning method," IEEE Trans. Syst., Man, Cybern., vol. SMC-18, Jan. /Feb. 1988.
9. H. Miyamoto, M. Kawato, T. Setoyama, and R. Suzuki, "Feedback-error-learning neural network for trajectory control of a robotic manipulator, " Neural Networks, vol. 1, pp. 251-265, 1988.
10. W. T. Miller, III, "Sensor-based control of robotic manipulators using a general learning algorithm, " IEEE J. Robotics Automat., vol. RA-3, pp. 157-165, Apr. 1987.
11. K. Nam and T. Kuc, " An application of the CMAC to robot control," in Proc. '88KACC, pp. 999-1005, 1988.
12. W. T. Miller, III, R. P. Hewes, F. H. Glanz, and L. G. Kraft, III, " Real-time dynamic control of an industrial manipulator using a neural-network-based learning controller," IEEE Trans. Robotics Automat., vol. 6, Feb. 1990.
13. J. S. Albus, " A new approach to manipulator control: The cerebellar model articulation control (CMAC)," Trans. ASME, J. Dyn. Syst., Meas. Contr., vol. 97, pp. 220-227, Sept. 1975.
14. D. E. Rumelhart, J. L. McClelland, and the PDP Research Group, " Learning internal representations by error propagation, " Parallel Distributed Processing: Explorations in the Microstructures of Cognition, vol. 1, MA: the MIT Press, pp. 318-362, 1986.