특집

지능 로봇 시스템 제어 기술 응용 및 하드웨어 구현

정 슬, 김성수, 송덕희, 엄일용 (충남대학교 메카트로닉스 공학과, 지능시스템 및 감성공학 실험실)

I. 서 론

최근에 전세계적으로 지능 로봇에 대한 연구가 활발하다. 미국, 일본을 비롯한 유럽의 선진국들은 지능로봇 사업을 국가적으로 추진하는 과제의 하나로 정하고 추진하고 있다. 우리나라에서도 지능 로봇을 국가 전략 사업의 하나로 지정하여 과기부, 산자부, 정통부 등 3개 부처가 추진하고 있는 실정이다. 지능 로봇의 궁극적인 목적인 인간과 같이 감지하고 인식 및 판단하여 행동할 수 있는 시스템을 만드는 것이다. 지능로봇 시스템은 단순히 지능을 가진 로봇 시스템으로 정의할 수 있으나, 그 범위가 매우 넓다. 로봇의 종류도 많을 뿐더러 지능 알고리즘도 매우 다양하기 때문이다.

본 논문에서는 '지능로봇시스템의 제어기술과 하드웨어 구현'이란 소제목 하에 지능로봇 시스템의 판단 및 행동의 일부분을 다루고자 한다. 앞에서 언급한 지능인식 기술을 통해 주변한경에 대한 인지를 한 후 판단하고 행동하는 것이 따른다. 시스템이 지능적으로 판단하여 행동한 다는 것은 하위 계층에서 시스템의 제어와 밀첩한 관계가 있음을 알 수 있다. 따라서, 지능알고리즘의 대표적인 도구인 신경회로망과 퍼지를 사용하여 가장 기본적인 구조를 바탕으로 지능제어기술 부분을 다루고자 한다. 신경회로망은 학습능력이 있어 우수하고 퍼지는 경험에 의한 판단 능력이 매우 우수하다.

II. 신경회로망 제어 알고리즘

1. 신경회로망 제어기술

1980년대에 들어서서 전성기를 이룬 신경회로망은제어분야에 많이 적용되었는데 그 이유는 학습능력과 비선형 맵핑 능력이 있기때문이다. 신경회로망을 주제어기로 사용하여 비선형 시스템의 역동역학으로 학습하려는 무모한 노력이후, 보조 제어기로 사용하는 방법이 대두하여 보편화 되었다. 기존의 제어기, PD, PID와 같은 제어기는 그대로 두고 보조제어기로 사용하여 시스템의 비선형적인 불확실성을 보상하는 것이다[1-3]. Feedback error learning 방식이 그 대표적인 경우이며[1],

reference compensation technique 방식 또한 다른 구조지만 같은 목적을 이룬다[2]. 적응 제어 구조를 모방한PID 제어기 계인 튜닝 방식은 구조적으로는 위의 두 방식과 다르지만 궁극적으로 불확실성을 보상하는 같은 방식임을 알 수 있다[3]. 신경회로망은 학습능력 비선형 맵핑 능력과 같은 우수한 특성을 가진 반면, 병렬처리 연산이 많아 실시간 구현이 쉽지 않은 단점이 있다.

2. Feedback error learning 제어 방식

Feedback error learning(FEL) 제어방식은 feedforward 제어방식의 구조로 그림 1에 나타나 있다. 신경회로망의 출력이 제어입력에 더해지게 되는데 이는 궁극적으로 신경회로망이 시스템의 역동역학 제어를 가능하게 한다.

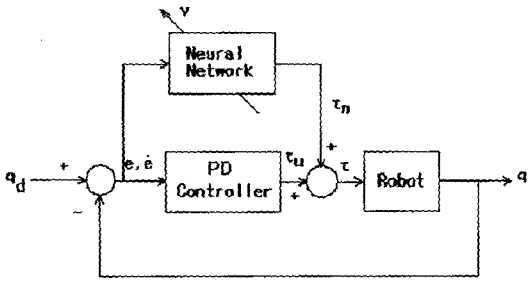


그림 1. FEL 신경회로망 제어방식

제어 입력 τ는 PD 제어기의 출력 τ_u와 신경망의 출력 τ_n의 합으로 다음과 같이 나타난다.

$$\tau = \tau_u + \tau_n \tag{1}$$

식(1)을 다시 정리하면 아래와 같다.

$$\tau_u = \tau - \tau_n \tag{2}$$

여기서, $e=q_d-q$ 이고 PD제어기는

$$\tau_u = K_P e + K_D \dot{e} \tag{3}$$

만약 PD 제어기의 출력 τ_u이 영으로 수렴하면, 즉 오차가 영으로 수렴하면, $\tau_u=\tau$이 되어 신경회로망의 출력이 시스템의 역동역학을 나타내게 된다.

따라서, FEL신경회로망 제어 방식의 경우, PD 제어기의 출력 τ_u이 영으로 수렴하도록 목적 함수를 설정한다. 목적함수는 다음과 같이 제어기의 출력으로 구성 한다.

$$E = \frac{1}{2}\tau_u{}^T \tau_u \tag{4}$$

따라서 역전파 알고리즘에 적용하기 위해 목적함수 E를 가중치 w로 편미분 하면 다음과 같다.

$$\begin{aligned}\frac{\partial E}{\partial w} &= \frac{\partial E}{\partial \tau_u}\frac{\partial \tau_u}{\partial w} \\ &= \tau_u \frac{\partial(\tau-\tau_n)}{\partial \tau_n}\frac{\partial \tau_n}{\partial w} \\ &= -\tau_u \frac{\partial \tau_n}{\partial w}\end{aligned} \tag{5}$$

이 그래디언트를 역전파 알고리즘에 적용한다.

최근에는 신경회로망으로 RBF(Radial basis function)넷을 사용하여 신경회로망 제어의 취약점인 시스템의 안정성을 검증하는 분석적인 연구가 활발하다.

3. Reference compensation technique 제어 방식

그림2는 RCT 기반의 신경회로망 제어기의 블록 선도를 나타낸다. PD 제어기의 출력에 신경회로망의 출력 신호를 더해서 보상하는 대신, 기준 입력에 보상한다. FEL 기반의 방

식과 구조적으로 비교해서 RCT 기반의 방식은 시스템의 내부를 수정하지 않고 외부에서 보상하는 장점이 있다.

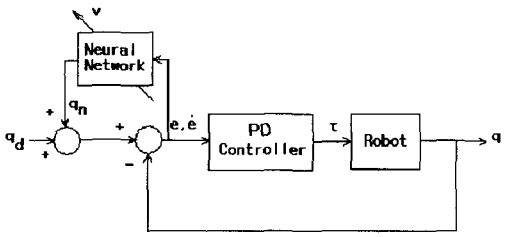


그림 2. RCT 신경회로망 제어방식

여기에서 PD 제어기의 출력 τ는 다음과 같다.

$$\tau = k_P e + k_D \dot{e}$$

$$= k_P(q_d - q + q_n) + k_D(\dot{q}_d - \dot{q} + \dot{q}_n) \quad (6)$$

$$= k_P \varepsilon + k_D \dot{\varepsilon} + k_P q_n + k_D \dot{q}_n$$

여기서 K_P, K_D는 PD제어기에 의해 결정되는 상수 값이고, $\varepsilon = q_d - q$이다.

토크를 τ라 하면,

$$k_P \varepsilon + k_D \dot{\varepsilon} + k_P q_n + k_D \dot{q}_n = \tau \quad (7)$$

정리하면,

$$k_P \varepsilon + k_D \dot{\varepsilon} = \tau - (k_P q_n + k_D \dot{q}_n) \quad (8)$$

신경망의 학습 신호를 다음과 같이 정의한다.

$$v = k_P \varepsilon + k_D \dot{\varepsilon} \quad (9)$$

신경망의 목적 함수는

$$E = \frac{1}{2} v^T v \quad (10)$$

목적함수를 가중치로 편미분 하면 다음과 같다.

$$\frac{\partial E}{\partial w} = \frac{\partial E}{\partial v} \frac{\partial v}{\partial w} \quad (11)$$

$$= -v\left(k_1 \frac{\partial q_n}{\partial w} + k_2 \frac{\partial \dot{q}_n}{\partial w}\right)$$

이 값을 역전파 알고리즘에 적용한다.

특히, RCT 제어구조는 보상 부분과 시스템 부분이 공간상으로 분리 가능하므로 무선으로 통신하는 로봇 시스템 제어에 유용하게 사용되어질 수 있다. 예를 들면, 자율주행 헬리곱터 제어의 경우 다음과 같이 지상에서의 보정을 통한 위치제어가 가능하게 된다.

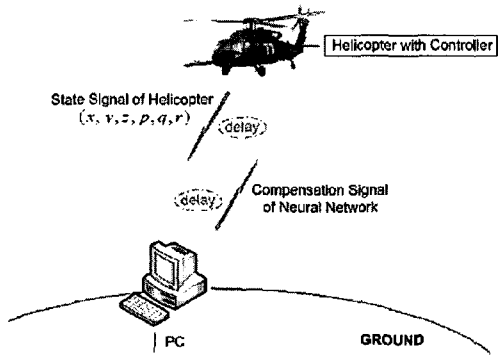


그림 3. 자유주행 헬리곱터의 신경회로망 제어방식

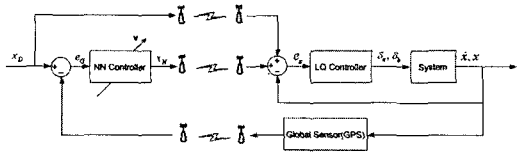


그림 4. 자율주행 헬리곱터의 RCT 신경회로망 제어

III. 퍼지 제어 알고리즘

1. 퍼지제어

퍼지는 Zadeh의 퍼지 논리(fuzzy logic) 발표 이후 Sugeno, Mamdani등을 거쳐 매우 활발한 연구가 이루어지고 있다. 특히 비선형이거나

복잡해서 정의하기 힘든 시스템에 대해서도 전문가의 경험에 의해 기존의 수식적 제어방식보다 더 잘 제어되는 경우가 많다. 신경회로망에 반해 퍼지 기술은 구현이 쉽고 연산시간이 오래 걸리지 않아 실시간 제어기로서의 많이 사용되어지고 있다.

퍼지 제어를 하기 위해서는 제어 변수와 이에 따른 소속 함수의 설정, 제어 규칙의 정의, 그리고 제어 변수의 정규화 과정 등이 필요하다. 그러나 퍼지 제어기의 설계에 있어 이러한 과정들이 결코 간단한 것만은 아니다. 시스템이 복잡해질수록 전문가의 의견이 서로 달라지고 이에 따라 소속 함수의 설정 및 제어 규칙의 정의 등이 설계자의 주관에 따라 달라질 수가 있다. 이는 제어기의 성능에 큰 차이를 가져오게 된다. 또한 이미 작성한 퍼지 제어기는 시스템의 변화에 능동적으로 대처하기가 어려운 단점이 있다. 즉, 시스템이 변화하게 되면 이에 맞춰 정규화 값을 변경해 주거나 제어기의 규칙을 수정해야 하는 번거로움을 안고 있다.

2. 퍼지 제어기의 응용사례

퍼지 이론은 가전용 세탁기에 퍼지 제어기를 적용하여 시판됨으로써 널리 알려지게 되었다. 이렇듯 퍼지 제어기는 실제 시스템에 적용되어 많이 사용되어지고 있다. 오른쪽 표는 퍼지 제어기를 사용하는 시스템과 회사를 보여준다. 많은 회사들이 다양한 분야에서 사용하고 있음을 알 수 있다.

표 1. 퍼지 제어기 응용 사례

퍼지 제어기 응용	회 사
에어컨	Hitachi, Matsushita, Mitsubishi, Sharp
aircraft contol	Rockwell Corp.
anti-lock brakes	Nissan
자동차 엔진	Nissan
자동차 트랜스 미션	Honda, Mitsubishi, Nissan Saturn, Subaru
복사기	캐논
cruise control	Isuzu, Nissan, Mitsubishi
elevator control	Fujitec, Mitsubishi Electric, Toshiba
식기세척기	Matsushita
전자레인지	Hitachi, Matsushita, Sanyo, Sharp, Toshiba
냉장고	Matsushita, Sharp
T.V	LG, Hitachi, Samsung, Sony
캠코더	Canon, Matsushita, Sanyo
세탁기	LG, Hitachi, Matsushita, Samsung, Sanyo, Sharp
기차 제어	Hitachi
space shuttle docking	NASA

3. PD 형태의 퍼지 제어기 설계

일반적인 PD 퍼지 제어기의 설계는 입력변수가 2개(오차, 오차의 미분)이고, 규칙 기반이 49이다. 최적화된 퍼지 제어기를 설계함에 있어 49개의 법칙을 모두 사용하지 않고 필요한 법칙만을 사용한다.

각 제어 변수들의 소속 함수는 그림 5와 같으며, 각 소속 함수는 모두 [-1, 1]에서 정규화 시킨 값을 사용한다.

위 소속 함수를 기반으로 제어 규칙을 설정하면 다음과 같이 나타낼 수 있다.

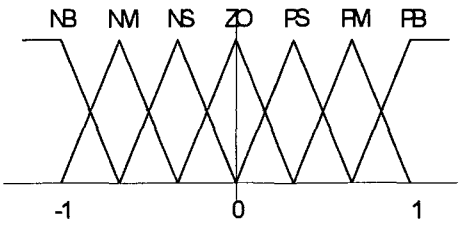


그림 5. 각 제어 변수의 소속 함수

R_θ : *If* e *is* A_i, $\dot{e}$ *is* B_i, *then* τ *is* C_i

따라서 이를 룰로 작성하면 표1.과 같은 테이블을 얻을 수 있다

표 2. 퍼지 규칙

τ		$\dot{e}$						
		NB	NM	NS	ZO	PS	PM	PB
e	NB	PB	PB	PM	PM	PM	PS	ZO
	NM	PB	PM	PM	PM	PS	ZO	NS
	NS	PM	PM	PM	PS	ZO	NS	NM
	ZO	PM	PM	PS	ZO	NS	NM	NM
	PS	PM	PS	ZO	NS	NM	NM	NM
	PM	PS	ZO	NS	NM	NM	NM	NB
	PB	ZO	NS	NM	NM	NM	NB	NB

ZO : Zero
PS(NS) : Positive(Negative) Small
PM(NM) : Positive(Negative) Medium
PB(NB) : Positive(Negative) Big

IV. 뉴로-퍼지 제어 알고리즘

1. 뉴로-퍼지 제어

일반적으로 퍼지 법칙을 작성하기 위해서는 그 시스템에 대한 전문적인 지식이나 작동 특성 등을 잘 알고 있어야 한다. 그렇지 않으면 법칙을 작성하는데 많은 어려움이 작용하게 된다.

이러한 단점들을 해결하기 위해 신경회로망을 결합한 뉴로-퍼지제어의 다양한 방법이 제안되었다. 그 중에서 신경회로망을 이용한 퍼지 규칙의 자동 생성 기법의 다층 신경회로망 구조가 그 축을 이룬다. 각 층은 퍼지화, 추론, 디퍼지화 등과 같은 각 퍼지 제어 방식의 진행과정을 나타내고, 이는 신경회로망과 퍼지제어의 장점만을 사용하여 시스템의 성능을 좋게 하는 결과를 보여 주었다. 그리고 기존의 퍼지제어 시스템에 신경회로망을 사용하여 보상하는 뉴로-퍼지제어 구조도 이루어 지고 있다.

2. FEL 기반의 신경회로망-퍼지 제어기

신경회로망-퍼지 제어기에 사용하는 모델은 역전파 알고리즘이며, 구조는 일반적인 feedforward 형태의 다층 구조를 사용하였다. FEL 방식의 시스템의 제어 구조는 그림 6에 나타나 있다. 신경회로망의 출력이 제어 입력에 더해진다. 앞의 그림1의 구조와 같으므로 같은 효과를 얻게된다.

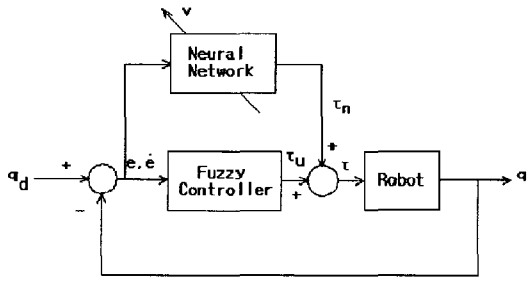


그림 6. FEL기반의 신경회로망과 퍼지 제어방식

3. RCT 기반의 신경회로망-퍼지 제어기

그림7은 RCT 기반의 신경회로망-퍼지 제어기의 블록 선도를 나타낸다. 퍼지 제어기의 입력에 신경회로망의 출력 신호를 더해서 보상하는 대신 기준 입력에 보상한다[5].

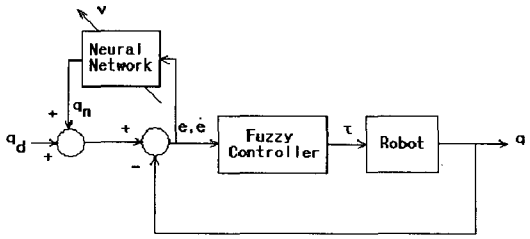


그림 7. RCT 기반의 신경회로망과 퍼지 제어방식

이 방식에서 신경회로망은 퍼지법칙을 변화시켜주는 효과를 얻게되어 보상하게 된다.

4. 신경회로망-퍼지 제어기

1) 뉴로-퍼지 제어기

그림8는 뉴로-퍼지 제어기의 일반적인 구조를 나타낸다. 여러 개의 층으로 구성되어 있으며, 각 층은 퍼지 과정을 나타낸다.

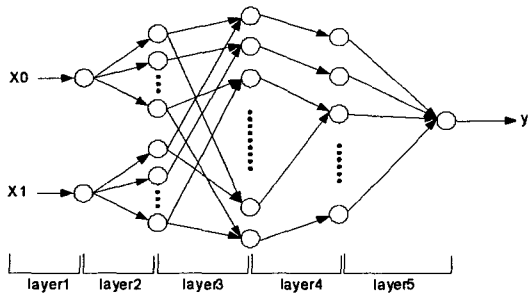


그림 8. 뉴로-퍼지 제어기의 네트워크 구조

Layer1은 입력 노드로 입력 언어 변수를 나타내고, layer2는 입력 언어 변수를 표현하기 위한 소속 함수를 나타낸다. Layer3는 규칙 노드로 퍼지 규칙을 표현한다. 그리고 layer4는 출력 언어 변수의 소속 함수이고 layer5는 출력 노드로서 각 출력 언어 변수를 표현한다. 또한 이 노드에서 학습 데이터가 네트워크로 입력되고, 실제 출력 값이 네트워크에서 출력 된다.

각 노드에서의 실행은 다음과 같다. Layer1에서는 단지 입력값을 다음 층으로 전달한다.

$$f^1 = u_i^1 \text{ 이고 } a^1 = f^1 \qquad (12)$$

여기서 u_i^1은 layer1에서의 i 번째 입력 언어 변수 x_i를 나타내며, a^1는 첫번째 층의 출력이 다음 층으로 전달되는 함수를 나타낸다.

Layer2에서는 종형 함수를 사용할 경우에 층의 출력 f^2는 다음과 같다.

$$f^2 = M_{x_i}^j(m_{ij}, \sigma_{ij}) = -\frac{(u_i^2 - m_{ij})^2}{\sigma_{ij}^2}$$
$$a^2 = e^{f^2} \qquad (13)$$

여기서, M_{ij}는 입력 언어 변수 x_i의 J번째 텀의 종형 함수의 중심값이고, σ_{ij}는 그 함수의 폭을 나타낸다. 그리고 u_i^2는 두 번째 층의 입력 값을 나타낸다.

Layer3에서는 퍼지 규칙의 전건부 결합을 수행하기 위해 사용된다. 따라서 fuzzy AND operation을 행하면 다음과 같다.

$$f^3 = \min(u_1^3, u_2^3, \cdots, u_p^3)$$
$$a^3 = f^3 \qquad (14)$$

Layer4에서는 동일한 결과를 갖는 규칙들을 통합하기 위해 fuzzy OR operation을 수행한다.

$$f^4 = \sum_{i=1}^{p} u_i^4$$
$$a^4 = \min(1, f^4) \qquad (15)$$

Layer5에서는 비퍼지화 방법인 무게 중심법을 실행한다.

$$f^5 = \sum w_{ij}^5 u_{ij}^5 = \sum (m_{ij}\sigma_{ij}) u_i^5$$
$$a^5 = \frac{f^5}{\sum \sigma_{ij} u_i^5} \qquad (16)$$

여기서 M_{ij}는 출력 언어 변수의 종형 함수의 중심값이고, σ_{ij}는 그 함수의 폭을 나타낸다.

2) TSK 뉴로-퍼지 제어기

TSK방식은 위의 뉴로-퍼지구조보단 간단한 구조이다. 출력부분이 간소화되어 각층을 정리하면 다음과 같다. 입력 변수를 x, y라 하

고 각 층의 출력을 f^l라 하자.

Layer1 퍼지층으로 아래와 같다.

$$f^1 = u_i^1 \tag{17}$$

종형 함수를 사용할 경우에 층의 출력 는 다음과 같다.

$$f^2 = M_{x_i}^j(m_{ij}, \sigma_{ij}) = -\frac{(u_i^2 - m_{ij})^2}{\sigma_{ij}^2} \tag{18}$$

Layer2에서는 멤버쉽 함수의 곱을 생성한다.

$$f_i^2 = w_i = \mu_{A_i}(x) \cdot \mu_{B_i}(y) \tag{19}$$

Layer3에서는 동일한 결과를 갖는 규칙들을 통합하기 위한 연산을 수행 한다.

$$f_i^3 = \bar{w}_i = \frac{w_i}{w_1 + w_2}, i = 1,2 \tag{20}$$

Layer4에서는 product를 실행한다.

$$f_i^4 = \bar{w}_i f_i = \bar{w}_i(p_i x + q_i y + r_i) \tag{21}$$

p_i, q_i, r_i는 설정해 주어야 하는 상수 값이다.

Layer 5에서는

$$f_i^5 = \sum \bar{w}_i f_i = \bar{w}_1 f_1 + \bar{w}_2 f_2 \tag{22}$$

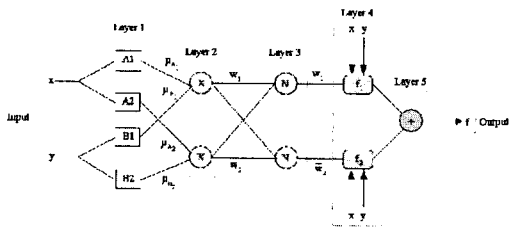


그림 9. TSK 방식의 뉴로-퍼지 제어구조

5. 시뮬레이션 연구

제어하고자 하는 로봇 모델은 그림 10과 같은 3축 로봇이다.

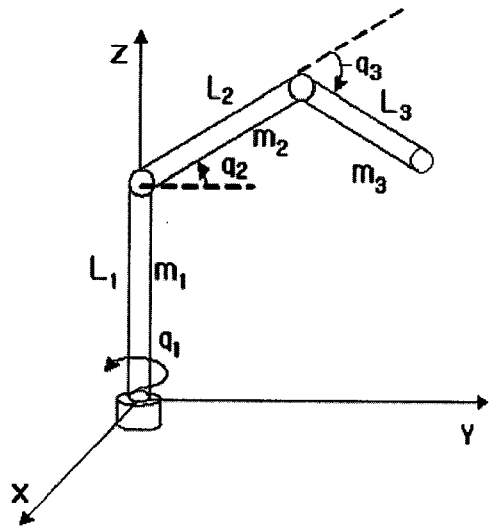


그림 10. 3축 로터리 로봇

일반적인 로봇 동력학식은 다음과 같다.

$$D(q)\ddot{q} + C(q,\dot{q})\dot{q} + G(q) + f_f(\dot{q}) = \tau \tag{23}$$

여기서, τ는 n×1의 토크 벡터, $D(q)$는 n×n의 관성행렬, $C(\dot{q}, q)\dot{q}$는 n×1의 코리올리스와 원심력 벡터, $G(q)$는 n×1의 중력 벡터, 그리고 $f_{f(\dot{q})}$는 n×1의 마찰력 벡터이다.

PD 제어기의 제어 법칙은 다음과 같다.

$$\tau = K_P e + K_D \dot{e} \tag{24}$$

그러므로 (23)과 (24)를 통해서 페루프 오차 방정식을 작성 하면 다음과 같이 나타난다.

$$\ddot{e} + D^{-1}K_P e + D^{-1}K_D \dot{e} = D^{-1}(D\ddot{q}_d + C + G + f) \tag{25}$$

로봇의 변수값은 각각 m1=15kg , L1=0.7m, m2=10kg, L2=0.6m, m3=5kg, L3=0.5m이다. 마찰 상수는

$$f_f = 15\dot{\theta}_1 + 10\dot{\theta}_2 + 5\dot{\theta}_3 \cdot$$

뉴로-퍼지 제어기를 사용한 경우로 그 출력 결과가 그림 11와 같이 나타난다. 잘 추종하는 것을 볼 수 있다.

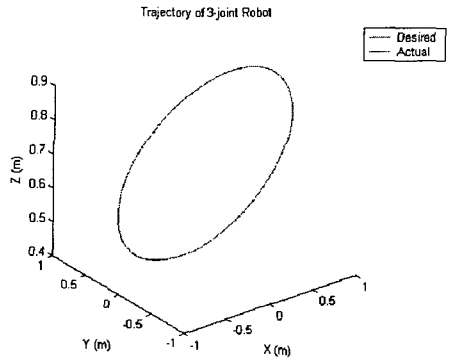


그림 11. 뉴로-퍼지 제어기의 위치 추종 성능

V. 신경망 하드웨어

1. 신경회로망의 하드웨어 구현 사례

퍼지제어기의 하드웨어 응용이 활발한 것과 마찬가지로 신경회로망의 하드웨어 응용도 활발하다. 퍼지제어기나 신경회로망 제어기 모두 전용칩으로의 상용화는 아직 활발하지 않다. 인텔에서도 신경회로망칩을 개발하였으나 더 이상 시판되지 않고 있고, 국내에서는 ETRI에서 개발하였으나 상용화 되지 않았다. 그 이유로는첫째, 고속의 마이크로프로세서의 출현으로 인한 컴퓨터의 속도가 빨라지고 있다는 것이다. 둘째로 가격경쟁력이 떨어진다는 것이다. 셋째, 신경칩에서 사용할 수 있는 뉴론의 수의 제한이다. 반면에 프로세서를 사용하면 프로그램적으로 뉴론의 수를 늘릴 수 있다. 넷째로 신경회로망 칩을 범용으로 사용하기 보다는 전문분야, 뇌나 눈과 같은 분야에서 사용하는 경우가 대부분이라는 것이다.

아래 그림은 파킨슨 병이나 알츠하이머 병과 같은 뇌 속의 신경세포의 손상을 해결하기 위해 인공신경칩을 사용하는 것을 보여준다. 뇌 속의 신경작용을 위한 칩을 나타내는데 가장 어려운 점은 실제 신경세포와 인공신경 세포간의 인터페이스이다.

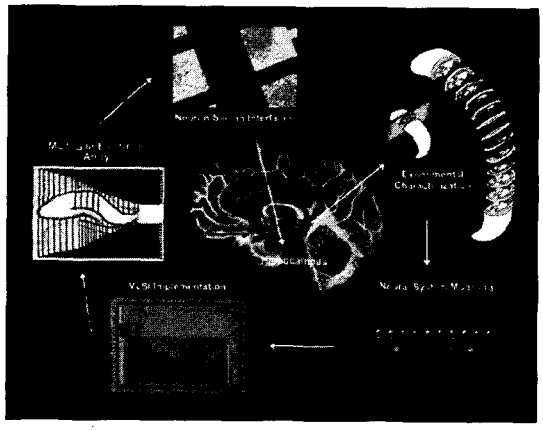


그림 12. 뇌속에서 신경 칩의 구현 개요

아래는 Neuricam사의 디지털 NC3003신경회로망 칩과 PCI 신경망 보드를 나타낸다. 주로 영상 처리 부분에 사용하고 있다.

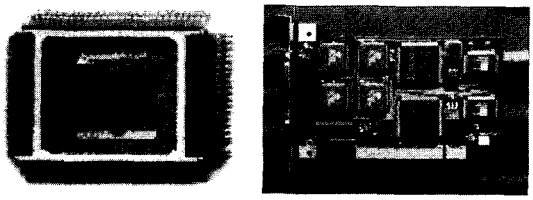


그림 13. Neuricam 사의 신경망 하드웨어

아래 그림은 IBM에서 개발한 ZISC(Zero Instruction Set Compter)를 나타내며 16개의 ZISC 칩으로 576개의 뉴론을 가지고 있다.

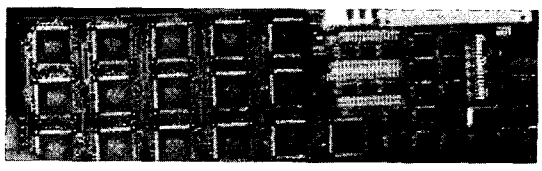


그림 14. 16 ZISC chips을 가진 ISA 카드

아래 그림은 CITIA의 32개의 뉴론을 가진 신경망 칩으로 뉴로-퍼지 제어를 위해 사용되어지고 있다.

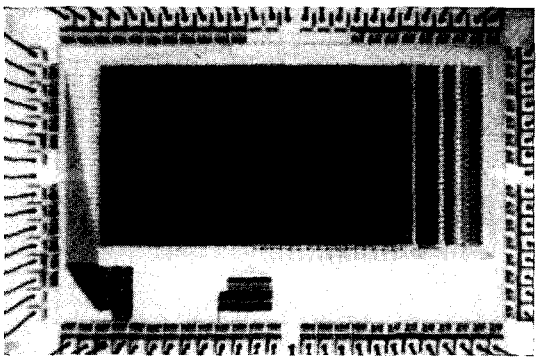


그림 15. CITIA 뉴론 칩

아래 그림은 CITIA의 새로운 칩 AMINAH로 MLP/RBF등의 신경회로망과 퍼지응용이 가능하다.

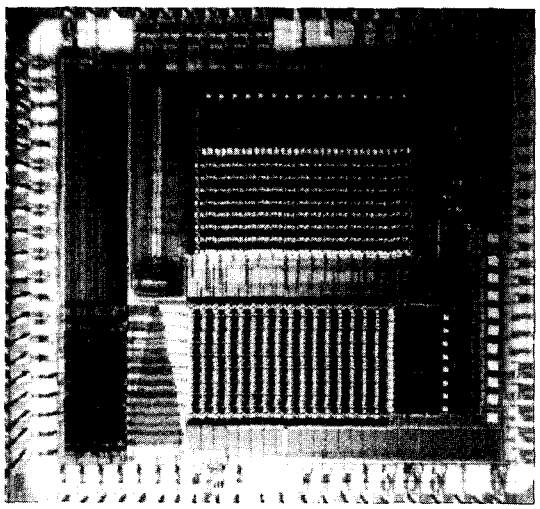


그림 16. CITIA 뉴로 퍼지 제어기 칩 AMINAH

이 밖에도 많은 신경망 칩이 있으나 실제적으로 상용화되어 널이 사용되는 칩은 드물다. 아직 연구 개발 단계에 머무르는 이유는 개발 가격이 비싸서 기존의 프로세서를 사용해서 저렴하게 구현할 수도 있기 때문이다. 다음은 신경망 칩을 사용하지 않고 기존의 프로세서를 사용해서 신경회로망 알고리즘을 구현한 사례를 중심으로 설명하고자 한다.

VI. 지능 제어기의 하드웨어 구현

1. 상용 시스템 기반의 구현

지능제어기를 가장 쉽게 구현하는 방법은 기존의 상용 패키지(DSP보드, DAQ카드, 소프트웨어) 를 사용하는 방법이다. 상용 팩키지는 사용하기 쉽지만 값이 매우 고가이다. 아래 그림은 상용 시스템을 사용하여 지능제어기를 구현한 모습이다[6].

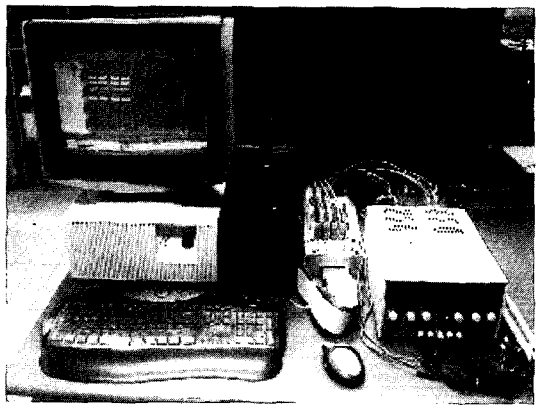


그림 17. 상용팩키지를 사용한 지능제어기

2. DSP-FPGA 기반의 구현

상용 팩키지를 사용하지 않고 지능제어기를 구현하기 위해서는 직접 설계하는 것이다. DSP 보드와 주변 모듈을 FPGA로 설계하여 DSP에 신경회로망을 구현하고 FPGA에

PID제어기 및 주변 요소들을 구현하면 저렴한 지능제어기를 구현할 수 있다. 지능제어기의 기본 블록 다이아 그램은 다음과 같다[7].

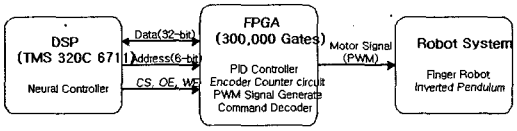


그림 18. DSP와 FPGA를 사용한 지능제어기 블록도

FPGA 제어기에 구현된 각 모듈은 다음과 같다.

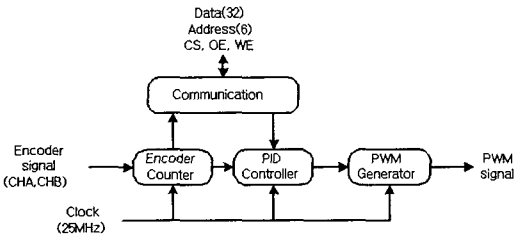


그림 19. FPGA를 내부구조

사용된 하드웨어는 DSP는 TI사의 TMS320 C6711이고 FPGA는 Altera사의 APEX II EP20K 300EQC240이며 , 그림 20과 같이 구성되었다.

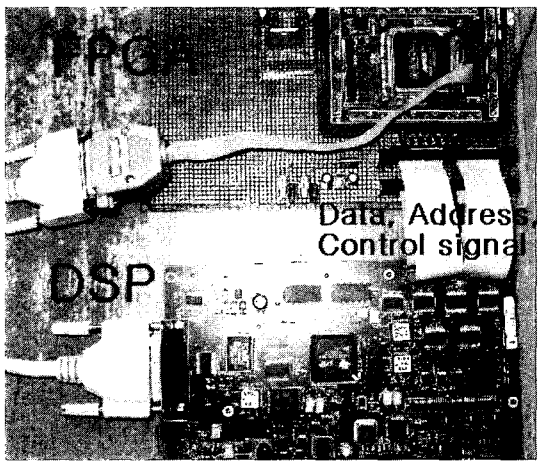


그림 20. DSP-FPGA 신경망 제어기 시스템 사진

3. MCU-FPGA 기반의 구현

좀더 하드웨어의 가격을 줄여보고자 DSP를 마이크로프로세서로 대체하여 지능 신경회로망 제어기를 구현하여 보았다. 마이크로프로세서는 삼성의 ARM 보드를 사용하였고 FPGA부분은 이전 것과 같이 사용하였다.

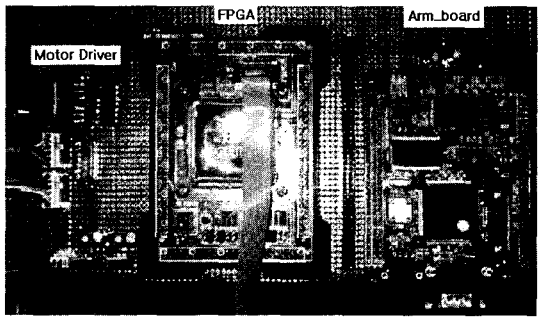


그림 21. MCU-FPGA 지능제어기

아래 그림은 MCU-FPGA지능제어기를 사용하는 역진자 시스템이다.

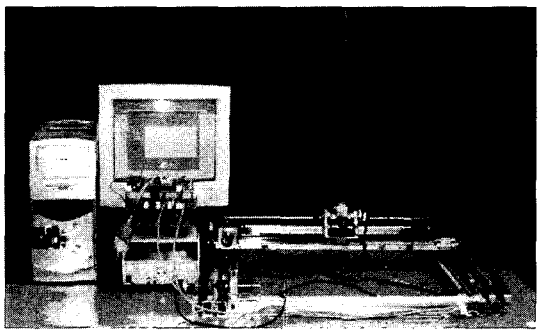


그림 22. 역진자 시스템

지능제어기의 역할은 역진자의 각도와 카트의 위치를 동시에 제어하는 것이다. 그림 19와 20은 제어된 결과이다. 그림 20에서 보면 카트가 주어진 경로를 잘 추종하는 것을 볼 수 있다.

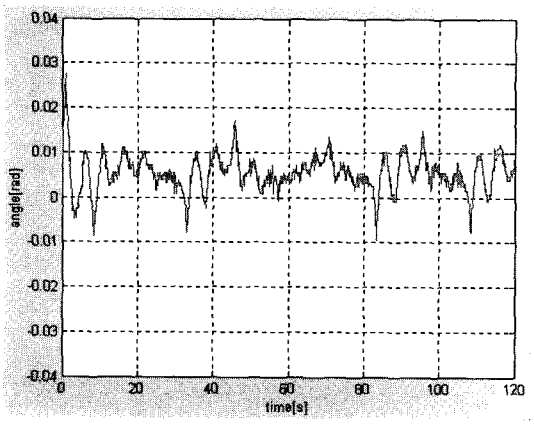


그림 23. 역진자 각도의 오차

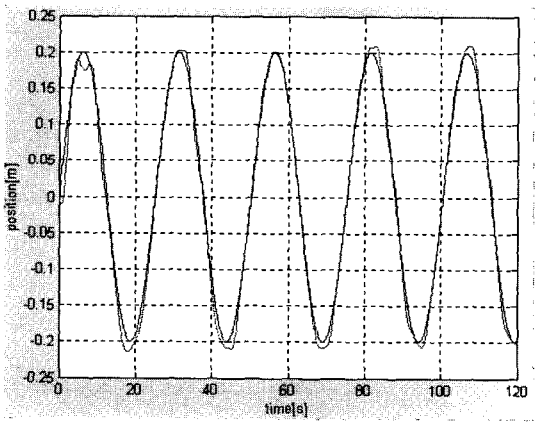


그림 24. 카트의 위치 추종

VII. 결 론

본 논문에서는 지능로봇 시스템의 제어 알고리즘을 대략적으로 소개하고 시뮬레이션 결과를 통해 알아 보았다. 신경회로망의 장점과 퍼지의 장점을 살려 결합한 지능제어기의 성능이 매우 우수함을 알아보았다. 실제로 신경회로망이나 퍼지의 경우 칩으로 개발된 사례가 있다. 하지만 실험실 차원에서 이를 구현하고자 할 경우 어려움이 있는 것이 사실이다. 따라서, 지능제어기의 하드웨어 구현에 있어서 신경회로망을 중심으로 실제로 구현된 예를 설명하였다. 실험 결과를 통해 지능제어기의 성능이 매우 우수함을 증명하였다.

앞으로는 하드웨어 기술이 발달함에 따라 간소화되고 연산 능력은 뛰어나, 지능제어기의 성능이 획기적으로 좋아질 것이다. 하지만 그에 따른 알고리즘의 연구가 병행되어야 할 것이다.

참고문헌

[1] M. Miyamoto, M. Kawato, T. Setoyama, and R. Suzuki, "Feedback error learning", *IEEE Trans. on Neural Networks*, vol.1, pp. 251-265, 1988

[2] S. Jung and T. C. Hsia, "A New Neural Network Technique for Robot Manipulators", *Robotica*, Vol. 13, pp. 477-484, 1995

[3] Sigeru Omatu, Marzuki Khalid and Rubiyah Yusof, "Neuro-Control and its Applications", *Springer*, 1996

[4] J. S. Wang, "Self-Adaptive Recurrent Neuro - Fuzzy Control of an Autonomous Underwater Vehicle", pp. 283-295, *IEEE Trans. on Robotics and Automations*, Vol. 19, No. 2, 2003

[5] Seul Jung and Duck Hee Song, "Neural Network Compensation Technique for Standard PD-Like Fuzzy Controlled Nonlinear Systems", pp.693-703, *IEEE CDC*, 2004

[6] S. Jung and H. Cho, "Balancing and Position Tracking Control of An Inverted Pendulum on An X-Y Plane Using Decentralized Neural networks," *IEEE/ASME Advanced Intelligent Mechatronics*,

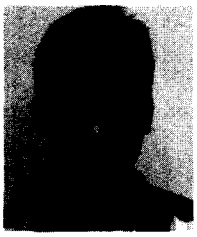
p.181-186, 2003.

[7] Sung su Kim and Seul Jung, "Hardware Implementation of a real time neural network controller with a DSP and an FPGA", *IEEE ICRA*, pp. 4639-4644, 2004

저자소개

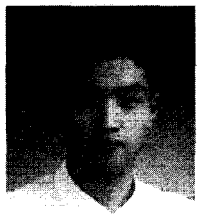
정 슬

1988년 미국 웨인주립대 전기 및 컴퓨터 공학과 졸업.
1991년 미국 캘리포니아대 데이비스 전기공학과 석사. 동대학 박사.
1997년~현재 충남대학교 메카트로닉스공학과 부교수.
주관심 분야 지능 제어 알고리즘 및 하드웨어 구현, 로봇의 인간의 지능적인 상호 작용, 필드 로봇 시스템.



김 성 수

2001년 경일대학교 제어계측공학과 졸업.
2003년 충남대학교 메카트로닉스공학과 석사.
주관심 분야
S.o.C 제어기 설계, DSP 및 마이크로프로세서 응용, 로보틱스



엄 일 용

2002년 충남대학교 메카트로닉스공학과 졸업.
현재 동대학 석사과정.
주관심 분야
지능제어, 헬리곱터 자율주행

송 덕 희

2002년 충남대학교 메카트로닉스공학과 졸업.
현재 동대학 석사과정.
주관심 분야
지능제어, 다관절 로봇