〈論 文〉 3-3-2

자기학습형 퍼지제어기를 이용한 유도전동기의 속도제어

朴 英 珉, 金 德 憲, 金 淵 忠, 金 才 文, 元 忠 淵

Speed Control of Induction Motor Using Self-Learning Fuzzy Controller

Young-Min Park, Duk-Heon Kim, Yuen-Chung Kim, Jae-Mun Kim, Chung-Yuen Won

요 약

본 논문은 신경회로망에 의한 퍼지제어기의 소속함수를 자동동조하는 방법을 제시하였다. 신경회로망 에뮬레이터는 퍼지제어기의 소속함수와 퍼지규칙을 재구성하는 경로를 제공하며, 재구성된 퍼지제어기는 유도전동기의 속도제어를 위해 사용한다. 따라서, 연산 시간과 시스템 성능의 관점에서 제안된 방법은 전동기 상수가 변동될 시에도 기존의 제어 방식보다 우수하다. 공간전압벡터 PWM 발생을 위한 고속연산을 수행하고 자기학습형 퍼지제어기 알고리즘을 구현하기 위해서 32비트 마이크로프로세서인 DSP(TMS320C31)을 사용하였다. 컴퓨터 시뮬레이션과 실험 결과를 통하여, 제안된 방식이 PI 제어기나 기존의 퍼지제어기보다 향상된 제어 성능을 보일 수 있음을 확인하였다.

ABSTRACT

In this paper, an auto-tuning method for fuzzy controller's membership functions based on the neural network is presented. The neural network emulator offers the path which reforms the fuzzy controller's membership functions and fuzzy rule, and the reformed fuzzy controller uses for speed control of induction motor. Thus, in the case of motor parameter variation, the proposed method is superior to a conventional method in the respect of operation time and system performance. 32bit micro-processor DSP(TMS320C31) is used to achieve the high speed calculation of the space voltage vector PWM and to build the self-learning fuzzy control algorithm. Through computer simulation and experimental results, it is confirmed that the proposed method can provide more improved control performance than that PI controller and conventional fuzzy controller.

Key Words: Neural Network, Neural Emulator, Fuzzy Rule, DSP, Induction Motor, Self Learning Fuzzy Control Algorithm

1. 서 론

비선형 특성을 가지고 있는 유도전동기는 PI 제어기를 사용할 때 제어이득 선정이 어렵고 최적의 이득에서도 파라미터 변동시에는 만족할 만한 제어특성을 얻기 어렵기 때문에 고성능 운전을 위하여 새로운 제어기법에 대한 연구가 활발히 진행되고 있다. 퍼지 이론은 기존의 제어 방식으로는 해결하기 어렵거나 시스템의 모델이 복잡한 경우에도 구현이 비교적 간단하여 실제 산업 분야에서 응용되고 있다[1][2]. 그러나, 퍼지제어기는 제어규칙, 소속함수, 정규화 및 비정규화 상수를 결정하기 위한 일반적인 규칙이 체계화되어 있지 않고, 전문가의 직관이나 경험에 의존하는 시행착오적인 제어기 설계가 이루어지는 문제점이 있다.

이러한 문제점으로 인하여 1980년대 말부터 퍼지 이론과 신경회로망 이론을 결합하여 상호 단점을 보완하고자 하는 노력이 활발히 이루어져 왔다. 퍼지추론과 신경회로망의 유사성은 퍼지추론의 최소-최대 연산(Min-Max Operation)이 신경회로망의 곱셈-덧셈 연산(Product-Sum Operation)에

해당되며, 또한 부분적인 특성함수인 소속함수를 시그모이드 함수에 의해 시스템 전체의 복잡한 비선형성을 표현한다. 그리고, 퍼지 추론이 논리구조를 취급하는 반면에, 신경회로망은 학습기능을 가지는 상호 보완 관계가 있다[4~6].

종래의 오프 라인 학습형 퍼지-신경회로망 제어 시스템의 문제점을 해결하기 위하여 본 논문에서는 실시간 제어와 실시간 플랜트 동정이 가능하고, 학습속도가 빠르며, 학습시 과도특성이 우수한 유도전동기 속도제어용 자기학습형 퍼지 제어 시스템을 구현하고자 한다. 자기학습형 퍼지 제어 시스템은 속도 제어를 위한 퍼지-신경회로망 제어기와 유도전동기 시스템을 모의하기 위한 신경회로망 에뮬레이터로 구성된다. 그리고, 퍼지-신경회로망을 온라인 실시간 제어기로써 이용하기 위해서는 매 샘플링 시간마다 제어기 출력단에서의 오차항을 실시간으로 계산하여야 한다.

이를 위해서는 실시간으로 플랜트의 입출력 관계를 모의하고 이를 통하여 플랜트 출력단의 오차항을 역전파시킬 수 있는 신경회로망 에뮬레이터의 구성이 필수적이다[6][7]. 또한, 유도전동기의 실시간 속도제어를 위해서는 학습시간이 짧고 학습시 과도특성이 우수하며 간단한 구조의 제어기가 요구된다.

본 논문에서는 유도전동기 속도제어를 위하여 실시간 플랜트 동정이 가능하도록 하기 위한 신경회로망 에뮬레이터를 이용하여 퍼지제어기의 소속함수와 제어규칙을 실시간으로 학습시키는 자기학습형 퍼지제어기를 사용하였다.

제안된 제어기의 우수성을 확인하기 위해 기존의 제어방법과 제안된 제어방법의 특성을 시뮬레이션과 실험을 통해 비교, 분석하여 그 우수성을 확인하였다.

2. 자기학습형 퍼지제어기의 구성

유도전동기 속도제어에 이용된 자기학습형 퍼지제어기는 제어규칙을 자동 생성하고 소속함수를 자동동조하기 위하여 퍼지제어기와 신경회로망 이론을 결합한 퍼지-신경회로망 제어기를 구성한다. 신경회로망 형태로 구성된 퍼지 제어규칙들은 다음과 같은 제어대상에 적합한 보통값(crisp value)의 결론부를 가지는 "IF ~ THEN ~" 형태를 가진다.

R_1 : IF x_1 is A_1 AND x_2 is B_1 THEN u is C_1
OR
R_2 : IF x_1 is A_2 AND x_2 is B_2 THEN u is C_2
OR
⋮
OR
R_n : IF x_1 is A_n AND x_2 is B_n THEN u is C_n

여기서, x_1, x_2는 제어입력, u는 제어출력이며 A_i, B_i는 범종형 소속함수를 갖는 입력 퍼지변수이고, C_i는 싱글톤형 소속함수를 갖는 출력 퍼지변수를 나타낸다.

그림 1은 자기학습형 퍼지제어기를 나타낸 것으로 퍼지제어기의 퍼지화에 해당하는 입력층, 제어규칙을 구성하는 은닉층 및 비퍼지화에 해당하는 출력층의 3개층 구조를 갖는 다층 퍼셉트론으로 구성된다. 샘플링 기간 k에서 자기학습형 퍼지제어기의 입력 $x_1(k)$는 기준속도 $\omega^*(k)$와 전동기 실제속도 $\omega(k)$와의 오차로 표시된다.

$$x_1(k) = \omega^*(k) - \omega(k) \tag{1}$$

그리고, 오차의 변화율 $x_2(k)$는

$$x_2(k) = x_1(k) - x_1(k-1) \tag{2}$$

로 나타내고, 각 층의 뉴런들은 상호 연결되어 은닉층과 출력층 사이의 연결강도는 가중치 W_i로 표시되고 있다. 입력층에서는 각 입력 변수의 소속함수 형태를 자동동조하기 위하여 소속함수값에 해당하는 $A_i(x_1)$과 $B_i(x_2)$를 출력한다.

입력에 대한 퍼지변수의 소속함수값은

$$A_i(x_1) = \frac{1}{1+\left(\frac{x_1 - a_{i2}}{a_{i1}}\right)^2} \tag{3}$$

$$B_i(x_2) = \frac{1}{1+\left(\frac{x_2 - b_{i2}}{b_{i1}}\right)^2} \tag{4}$$

이 값들은 은닉층으로 전달되며, 은닉층에서는 이 두 값을 곱한 값

$$\mu_i = A_i(x_1) \times B_i(x_2) \tag{5}$$

를 출력하며 가중치 W_i와 곱해져 출력층으로 전달한다. 여기서, μ_i는 퍼지제어기의 소속함수값이다.

출력층은 비퍼지화 단계에 해당하므로 식 (6)과 같이 무게중심법에 의해 비퍼지화를 실행한다.

$$u = \frac{\sum_i^n \mu_i W_i}{\sum_i^n \mu_i} \tag{6}$$

자기학습형 퍼지제어기의 소속함수 및 제어규칙은 학습을 통하여 동조된다. 다층 퍼셉트론의 학습방법으로는 주로 일

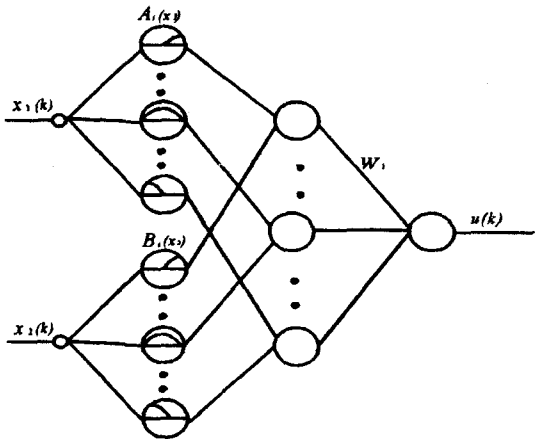


그림 1 자기학습형 퍼지제어기
Fig. 1 Self-learning fuzzy controller

반화된 델타 규칙을 이용한 오차 역전파 알고리즘(Back Propagation Algorithm)이 많이 사용되고 있다. 이 방법은 지도학습의 일종으로 지도출력과 실제값 사이의 평균 제곱 오차값을 최소화하는 최소자승법을 사용한다. 하지만, 자기학습형 퍼지제어기를 이용하여 유도전동기 속도제어기를 구성하였을 경우 학습을 위한 제어기 출력단에서의 오차항은 지도출력이 존재하지 않으므로 평균 제곱 오차값을 구할 수 없다.

본 논문에서는 자기학습형 퍼지제어기의 지도출력을 구하지 않고, 신경회로망 에뮬레이터를 통하여 유도전동기 출력단에서의 오차항을 역전파함으로써 지도출력값을 구하였다. 결국 신경회로망 에뮬레이터의 역할은 유도전동기의 전방향 동특성(Forward Dynamics)을 학습하고 유도전동기 출력단에서의 오차항을 역전파시켜 제어기 출력단에서의 오차항을 계산할 수 있는 오차 역전파 경로를 제공하는 것이다.

한편, 신경회로망 에뮬레이터는 그림 2와 같이 세개의 입력층 뉴런, 다섯개의 은닉층 뉴런, 한개의 출력층 뉴런을 가지

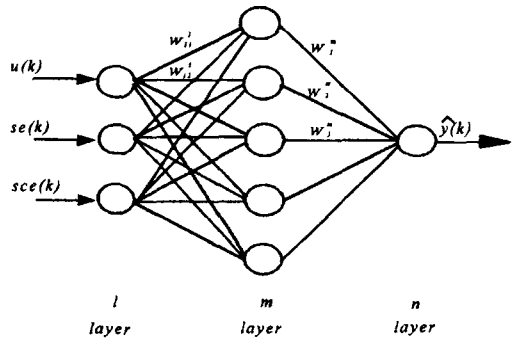


그림 2 신경회로망 에뮬레이터
Fig. 2 Neural network emulator

는 3층의 다층 퍼셉트론으로 구성된다.

신경회로망 에뮬레이터의 입력으로는 자기학습형 퍼지제어기의 출력값 $u(k)$, 제어기의 입력으로 사용된 기준속도와 실제속도의 오차 $se(k)=x_1(k)$, 그리고 오차의 변화율 $sce(k)=x_2(k)$을 선택하였다. 이렇게 하면 신경회로망의 구조가 간단하게 되어 연산시간이 짧아지며, 학습방법으로 일반화된 오차 역전파 알고리즘의 이용이 가능해진다. 따라서, 하드웨어를 구성할 경우 샘플링 기간을 줄일 수 있다. 또한, 에뮬레이터 입력층의 입력범위가 ±1의 값을 가지므로 활성함수로서 양방향 시 그모이드 함수인 tanh(·)를 사용하였다.

자기학습형 퍼지제어기와 신경회로망 에뮬레이터를 사용한 유도전동기 속도제어 시스템은 그림 3과 같다.

신경회로망 에뮬레이터를 학습시키기 위한 학습 과정을 살펴보면, 먼저 신경회로망 에뮬레이터를 학습시키기 위하여 에뮬레이터의 출력과 유도전동기의 실제 출력과의 오차항 $\hat{e}(k)$를 구하면 다음과 같다.

$$\hat{e}(k)=\frac{1}{2}\left(y(k)-\hat{y}(k)\right)^2 \tag{7}$$

여기서, $y(k)$는 유도전동기의 실제속도이며, $\hat{y}(k)$는 신경회로망 에뮬레이터의 출력이다. 학습은 식 (7)을 최소화하도록 다음과 같이 일반화된 델타 규칙에 의해 이루어진다[3].

$$\Delta W_j^m \propto -\frac{\partial \hat{e}(k)}{\partial W_j^m} \tag{8}$$

여기서, W_j^m은 에뮬레이터에서의 m번째 층의 j번째 뉴런과 다음 층과의 연결 가중치를 나타낸다. 위 식의 우변은 체인 규칙(chain rule)에 의해 다음과 같이 표현된다.

$$\frac{\partial \hat{e}(k)}{\partial W_j^m}=\frac{\partial \hat{e}(k)}{\partial net^n}\frac{\partial net^n}{\partial W_j^m} \tag{9}$$

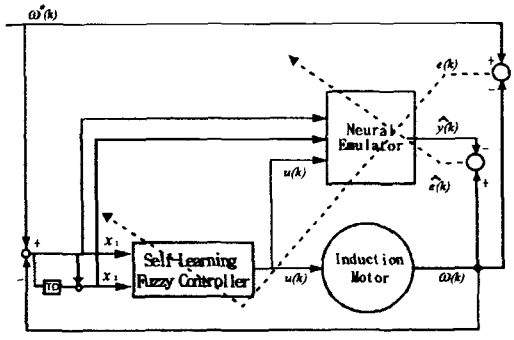


그림 3 자기학습형 퍼지제어기를 이용한 유도전동기 속도제어 시스템
Fig. 3 Speed control system of induction motor using self-learning fuzzy controller

일반적인 뉴런의 입력합(net)과 입력합에 의한 뉴런의 출력(out)에 의하여 식 (9)에서의 우변 두번째 항은 다음과 같이 표현할 수 있다.

$$\frac{\partial net^n}{\partial W_j^m} = \frac{\partial}{\partial W_j^m}\sum_j out_j^m W_j^m = out_j^m \tag{10}$$

여기서, net^n는 n번째 층의 입력합이고, out_j^m는 m번째 층의 j번째 뉴런의 출력을 나타낸다. δ^n을 다음과 같이 정의하면

$$\delta^n = \frac{\partial \hat{e}(k)}{\partial net^n} \tag{11}$$

식 (9)는 다음과 같다.

$$\frac{\partial \hat{e}(k)}{\partial W_j^m} = \delta^n out_j^m \tag{12}$$

그러므로, $\hat{e}(k)$에 대한 기울기가 감소되기 위해서는 가중치 조절은 다음과 같이 이루어져야 한다.

$$\Delta W_j^m = \eta \delta^n out_j^m \tag{13}$$

여기서, $\eta(0<\eta<1)$ 는 학습률이다.

δ^n는 n번째 층의 뉴런의 오차항이며, 다음과 같이 구해진다.

$$\delta^n = \frac{\partial \hat{e}(k)}{\partial net^n} = \frac{\partial \hat{e}(k)}{\partial out^n}\frac{\partial out^n}{\partial net^n} \tag{14}$$

뉴런의 출력함수를 정의하면 다음과 같다.

$$out^n = f(net^n) = \tanh(net^n) \tag{15}$$

식 (14)에서 우변의 두 번째 항은 다음과 같이 표현된다.

$$\frac{\partial out^n}{\partial net^n} = f'(net^n) \tag{16}$$

결국 식 (7)을 이용하여 출력층의 뉴런에서의 오차항은 다음과 같다.

$$\begin{aligned}\delta^n &= f'(net^n)(y(k)-\hat{y}(k)) \\ &= (1-\hat{y}(k)^2)(y(k)-\hat{y}(k))\end{aligned} \tag{17}$$

은닉층의 경우에는 지도출력값을 알 수 없으므로 이미 알고 있는 출력층에서의 오차항을 이용하면 다음과 같이 구할 수 있다.

$$\begin{aligned}\frac{\partial \hat{e}(k)}{\partial out_j^m} &= \frac{\partial \hat{e}(k)}{\partial net^n}\frac{\partial net^n}{\partial out_j^m} \\ &= \frac{\partial \hat{e}(k)}{\partial net^n}\frac{\partial}{\partial out_j^m}\sum_j out_j^m W_j^m \\ &= \frac{\partial \hat{e}(k)}{\partial net^n} W_j^m = \delta^n W_j^m\end{aligned} \tag{18}$$

위의 결과를 식 (17)에 대입하면 은닉층의 경우 오차항은 다음과 같이 표시된다.

$$\begin{aligned}\delta_j^m &= f'(net_j^m)\delta^n W_j^m \\ &= \left(1-out_j^m\right)^2)\delta^n W_j^m\end{aligned} \tag{19}$$

일반적으로 보다 빠른 학습속도를 위해 식 (13)에 관성(momentum)항이 추가되는데 이 경우 가중치 조절은 다음과 같이 표현된다.

$$\Delta W_j^m(k+1) = \eta \delta^n out_j^m + \alpha \Delta W_j^m(k) \tag{20}$$

$$W_j^m(k+1) = W_j^m(k) + \Delta W_j^m(k+1) \tag{21}$$

여기서, k는 학습 반복 횟수, $\alpha(0<\alpha<1)$는 관성항이다.

구성된 신경회로망 에뮬레이터가 플랜트 출력단에서의 오차항을 정확히 역전파시키기 위해서는 에뮬레이터가 플랜트의 입출력을 잘 모의할 수 있도록 하는 예비 학습 단계가 필요하다. 이를 위하여 본 논문에서는 한 주기의 기준입력 기간 동안 임의의 제어 입력 패턴을 1회 인가하여 예비 학습을 실시하였다.

자기학습형 퍼지제어기는 제어대상이 주어진 기준값을 추종할 수 있도록 즉, $x_1(k)\simeq 0$이 되도록 제어입력값 $u(k)$를 생성하고 있다. 제어기를 학습시키기 위해서는

$$ue(k) = (u_t(k)-u(k)),\ u_t(k)\text{: 지도출력값} \tag{22}$$

의 계산이 필요하지만 원하는 출력을 얻기 위한 제어입력 $u_t(k)$는 계산이 불가능하므로 신경회로망 에뮬레이터를 이용하여 $u_t(k)$를 구한다. 즉, 먼저 플랜트 출력단에서의 오차항을 구하고 이를 신경회로망 에뮬레이터를 통하여 역전파시켜 에뮬레이터 입력층에서 자기학습형 퍼지제어기와 연결된 노드의 오차항을 이용하면 된다.

자기학습형 퍼지제어기의 학습은 오차 역전파 알고리즘을 이용하며 학습과정은 다음과 같다. 먼저, 퍼지제어기 출력층

에서의 가중치 변화량은

$$\Delta W_i \propto -\frac{\partial ue(k)}{\partial W_i(k)} \tag{23}$$

체인 규칙에 의해

$$\frac{\partial ue(k)}{\partial W_i} = \frac{\partial ue(k)}{\partial net^k}\frac{\partial net^k}{\partial W_i} \tag{24}$$

여기서, net^k 는 제어기 출력층 뉴런의 입력합이며 다음 식과 같다.

$$net^k = \sum \mu_i W_i \tag{25}$$

식 (24)의 우변 두 번째 항은

$$\frac{\partial net^k}{\partial W_i} = \frac{\partial}{\partial W_i}\sum_i out_i^j W_i = out_i^j \tag{26}$$

이 되며, 여기서 out_i^j 은 은닉층의 출력값을 의미한다. 다음을 정의하면

$$\delta^k = \frac{\partial ue(k)}{\partial net^k} = \frac{\partial ue(k)}{\partial u(k)}\frac{\partial u(k)}{\partial net^k} \tag{27}$$

이 되며,

$$\frac{\partial u(k)}{\partial net^k} = \frac{1}{\sum \mu_i} \tag{28}$$

과 같이 계산되지만, $ue(k)$를 알지 못하므로 $\frac{\partial ue(k)}{\partial u(k)}$ 는 계산이 불가능하다. 따라서, 본 논문에서는 $\frac{\partial ue(k)}{\partial u(k)}$ 를 직접 계산하는 대신 신경회로망 에뮬레이터를 통하여 역전파된 오차값을 이용하였다. 퍼지제어기의 출력층에서의 오차항을 구하기 위하여 플랜트의 출력단의 오차항을 다음과 같이 정의할 수 있다.

$$e(k) = \frac{1}{2}x_1^2 \tag{29}$$

이 오차항은 역전파 알고리즘에 의해 신경회로망 에뮬레이터를 통하여 역전파되어 은닉층에서의 오차항

$$\begin{aligned}\delta_i^m &= f'\left(net_i^m\right)\delta^n W_i^m \\ &= \left(1-out_i^m\right)^2 \delta^n W_j^m\end{aligned} \tag{30}$$

$$\delta^n = f'\left(net^n\right)x_1 = \left(1-y(k)^2\right)x_1 \tag{31}$$

이 구해진다. 따라서, 신경회로망 에뮬레이터 입력층에서 자기학습형 퍼지제어기의 출력 $u(k)$와 연결된 노드에서의 오차항은 에뮬레이터 은닉층의 각 노드에서 역전파되어 온 오차항의 합이 된다.

$$\delta_j^l = \sum_i \delta_i^m \tag{32}$$

식 (32)의 오차항은 다시 퍼지 제어기를 학습시키기 위한 오차항으로 사용되며 $\frac{\partial ue(k)}{\partial u(k)}$ 를 에뮬레이터를 통하여 역전파된 오차항 식 (32)의 값으로 대치하면

$$\delta^k = \frac{\delta_j^l}{\sum \mu_i} \tag{33}$$

이 되며, 이것을 이용하여 제어기 출력단에서의 가중치 변화량은

$$\Delta W_i = -\eta\delta_j^l \frac{\mu_i(k)}{\sum \mu_i(k)} \tag{34}$$

으로 구해진다. 그리고, 은닉층에서의 오차항 역시 언급한 것과 같은 방법으로 유사하게 구해진다. 입력층의 소속함수 파라미터의 변화량 $\Delta a_{i1} \propto -\frac{\partial e(k)}{\partial a_{i1}(k)}$ 이므로

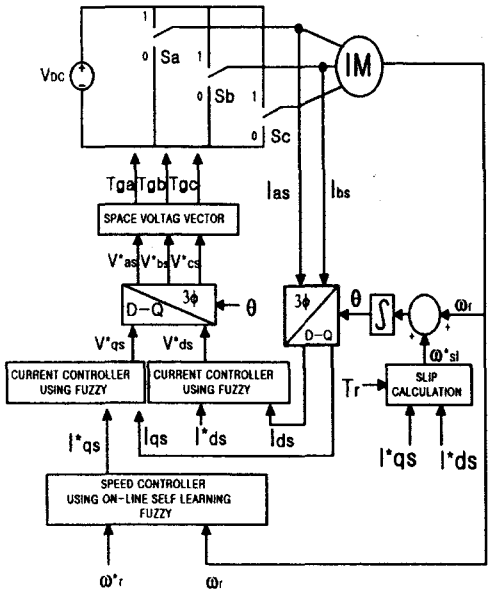


그림 4 간접벡터제어 유도전동기 속도제어 시스템
Fig. 4 Speed control of induction motor with indirect vector control

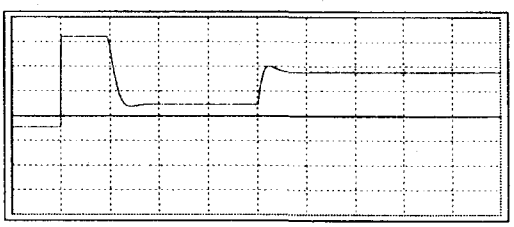
$$\Delta a_{i1} = -\eta \frac{\partial ue(k)}{\partial u(k)} \frac{\partial u(k)}{\partial \mu_i(k)} \frac{\partial \mu_i(k)}{\partial A_{i1}(k)} \frac{\partial A_{i1}(k)}{\partial a_{i1}(k)}$$
$$= -\eta \delta_j^l \frac{1}{\sum \mu_i(k)} \left(W_i(k) - u(k)\right) \times \quad (35)$$
$$B_i\left(x_2(k)\right) \frac{1}{a_{i1}(k)} \left[A_i(k) - A_i^2(k)\right]$$

$$\Delta a_{i2} = -\eta \frac{\partial ue(k)}{\partial u(k)} \frac{\partial u(k)}{\partial \mu_i(k)} \frac{\partial \mu_i(k)}{\partial A_{i1}(k)} \frac{\partial A_{i1}(k)}{\partial a_{i2}(k)}$$
$$= -\eta \delta_j^l \frac{1}{\sum \mu_i(k)} \left(W_i(k) - u(k)\right) \times \quad (36)$$
$$B_i\left(x_2(k)\right) \frac{1}{a_{i2}(k)} \left[A_i(k) - A_i^2(k)\right]$$

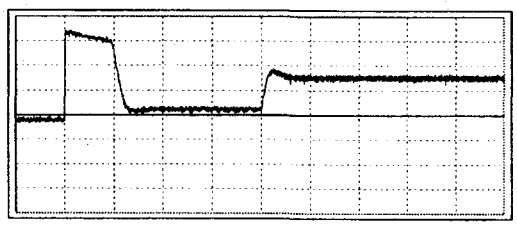
$$\Delta b_{i1} = -\eta \frac{\partial ue(k)}{\partial u(k)} \frac{\partial u(k)}{\partial \mu_i(k)} \frac{\partial \mu_i(k)}{\partial B_i(k)} \frac{\partial B_i(k)}{\partial b_{i1}(k)}$$
$$= -\eta \delta_j^l \frac{1}{\sum \mu_i(k)} \left(W_i(k) - u(k)\right) \times \quad (37)$$
$$A_i\left(x_1(k)\right) \frac{1}{b_{i1}(k)} \left[B_i(k) - B_i^2(k)\right]$$

$$\Delta b_{i2} = -\eta \frac{\partial ue(k)}{\partial u(k)} \frac{\partial u(k)}{\partial \mu_i(k)} \frac{\partial \mu_i(k)}{\partial B_i(k)} \frac{\partial B_i(k)}{\partial b_{i2}(k)}$$
$$= -\eta \delta_j^l \frac{1}{\sum \mu_i(k)} \left(W_i(k) - u(k)\right) \times \quad (38)$$
$$A_i\left(x_1(k)\right) \frac{1}{b_{i2}(k)} \left[B_i(k) - B_i^2(k)\right]$$

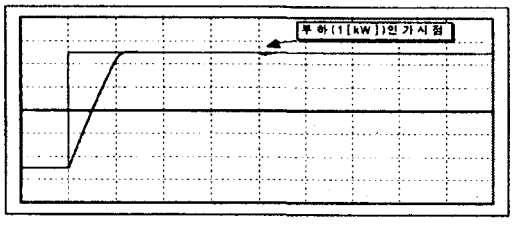
로 계산된다[3]~[5][8].



(X축: 0.5 sec/div, Y축: 2.5A/div)
(a) 토오크 성분 전류 지령치
(a) Reference current of torque component

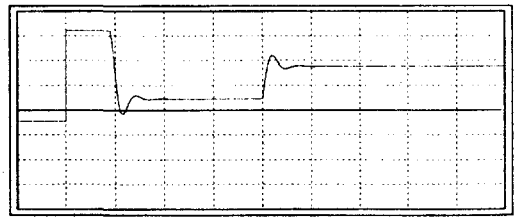


(X축: 0.5 sec/div, Y축: 2.5A/div)
(b) 토오크 성분 전류
(b) Real current of torque component

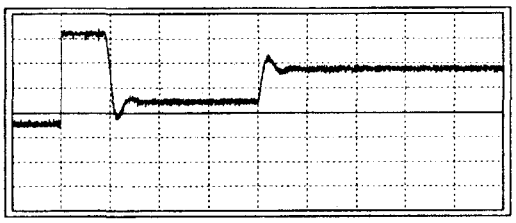


(X축: 0.5 sec/div, Y축: 400 rpm/div)
(c) 속도 응답
(c) Speed response for the reference change

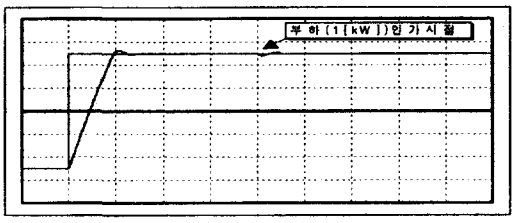
그림 5 PI 속도제어기와 PI 전류제어기의 응답
Fig. 5 Responses of PI speed controller and PI current controller



(X축: 0.5 sec/div, Y축: 2.5A/div)
(a) 토오크 성분 전류 지령치
(a) Reference current of torque component



(X축: 0.5 sec/div, Y축: 2.5A/div)
(b) 토오크 성분 전류
(b) Real current of torque component



(X축: 0.5 sec/div, Y축: 400 rpm/div)
(c) 속도 응답
(c) Speed response for the reference change

그림 6 퍼지 속도제어기와 퍼지 전류제어기의 응답
Fig. 6 Responses of fuzzy speed controller and fuzzy current controller

3. 시뮬레이션 결과

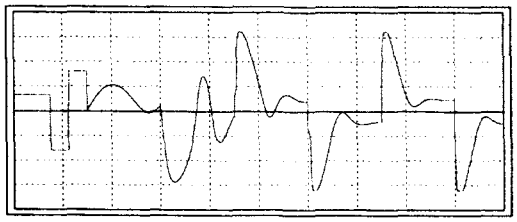
본 논문에서 제시한 제어 알고리즘의 효용성을 고찰하고, 제어 특성을 검토하기 위하여 실험에 앞서 C-언어를 사용하여 시뮬레이션하였다. 유도전동기의 속도제어는 PI 제어기, off-line 퍼지제어기, 그리고 실시간 자기학습형 퍼지제어기를 비교하여 실시하였고, 샘플링 시간은 3[ms]로 하였다. 전류제어는 PI 제어기와 look-up table로 구성된 퍼지제어기를 비교하여 실시하였으며, 매 100[㎲]마다 이루어진다.

그림 4는 시뮬레이션을 위하여 구성된 유도전동기 속도 및 전류제어 시스템의 구성도이다. 파라미터의 변동나 외란과 같은 제어환경의 변화에 대응할 수 있도록 학습기능을 가지고 있는 자기학습형 퍼지제어기를 비선형 시변계인 유도전동기 속도제어 시스템에 적용하여 실시간으로 제어이득을 자동 동조할 수 있도록 하였다[9,10].

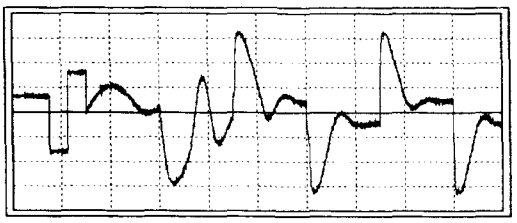
3상 유도전동기를 -1000~1000[rpm]으로 정역회전과 부하 인가 시뮬레이션을 PI 속도제어기, 퍼지 속도제어기, 자기학습형 퍼지 속도제어기와 PI 전류제어기, 퍼지 전류제어기와 비교하여 수행하였으며, 인가한 부하는 1[kW]이다.

그림 6은 퍼지 속도제어기와 퍼지 전류제어기를 사용했을 때의 토오크 성분 전류 지령치와 토오크 성분 전류, 그리고 유도전동기의 실제속도와 기준속도의 파형을 보여주고 있는데, 그림 5의 PI 속도 제어기와 PI 전류 제어기를 사용했을 때보다 오버슈트와 진동은 거의 비슷하고 부하에 대한 강인성은 우수한 특성을 보였다.

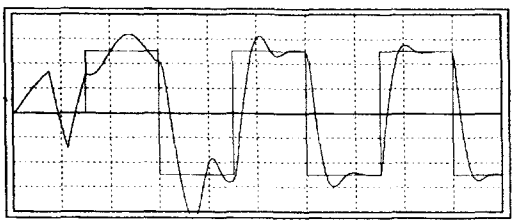
그림 7은 무부하 상태에서 일정시간 동안 임의의 제어량을 내 보낸 후 -1000~1000[rpm]으로 변화하는 기준속도를 인가했을 때, 자기학습형 퍼지제어기는 신경망 에뮬레이터의 학습기능을 이용하여 빠른 시간 내에 퍼지제어기의 소속함수와 제어규칙을 찾아가고 있음을 보여준다. 그림 8은 본 논문



(X축: 0.5 sec/div, Y축: 2.5A/div)
(a) 토오크 성분 전류 지령치
(a) Reference current of torque component

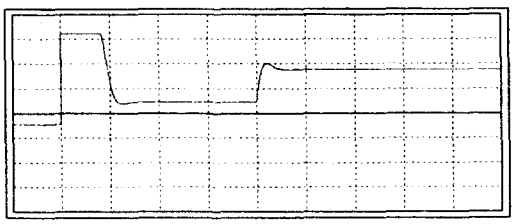


(X축: 0.5 sec/div, Y축: 2.5A/div)
(b) 토오크 성분 전류
(b) Real current of torque component

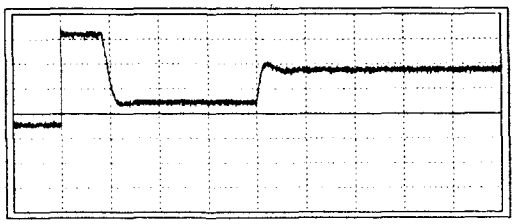


(X축: 0.5 sec/div, Y축: 400 rpm/div)
(c) 속도 응답
(c) Speed response for the reference change

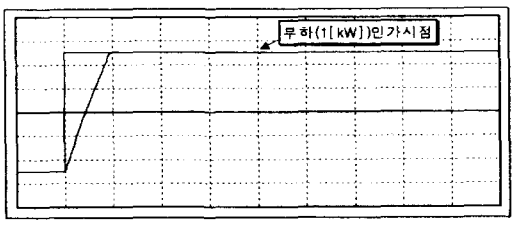
그림 7 자기학습형 퍼지제어기의 학습과정의 응답
Fig. 7 Responses of training of self-learning fuzzy controller



(X축: 0.5 sec/div, Y축: 2.5A/div)
(a) 토오크 성분 전류 지령치
(a) Reference current of torque component



(X축: 0.5 sec/div, Y축: 2.5A/div)
(b) 토오크 성분 전류
(b) Real current of torque component



(X축: 0.5 sec/div, Y축: 400 rpm/div)
(c) 속도 응답
(c) Speed response for the reference change

그림 8 자기학습형 퍼지 속도제어기와 퍼지 전류제어기의 응답
Fig. 8 Responses of self-learning fuzzy controller and fuzzy current controller

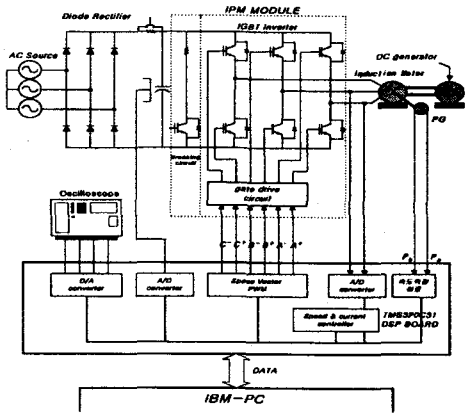


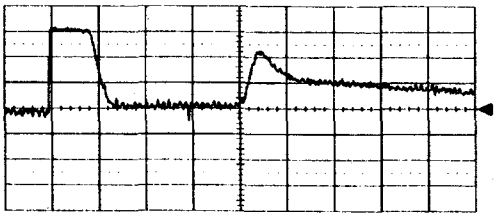
그림 9 실험장치 구성도
Fig. 9 Block diagram of experimental setup

에서 제안한 5회 학습 후의 자기학습형 퍼지 속도제어기와 퍼지 전류제어기를 사용한 것으로써 그림 6의 off-line으로 구성된 퍼지 속도제어기보다 정역운전에 대한 속도응답과 부하변화에 대하여 강인하며, 그림 5의 PI 전류제어기와 비교하면 오버슈트와 진동은 비슷하나 응답시간과 부하변화에 대한 강인성면에서는 우수하였다.

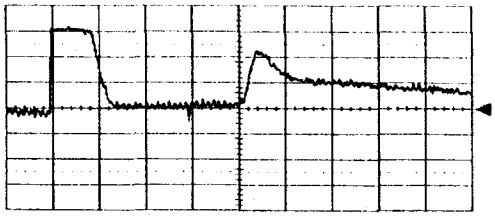
유도전동기 속도제어에서는 PI 제어기, 퍼지 제어기, 자기학습형 퍼지제어기 중 자기학습형 퍼지제어기가 우수하며, 전류제어에서는 PI 제어기, 퍼지 제어기 중 퍼지 제어기가 토오크 맥동 저감면에서 우수한 제어성능을 보여 준다.

4. 실험결과 및 검토

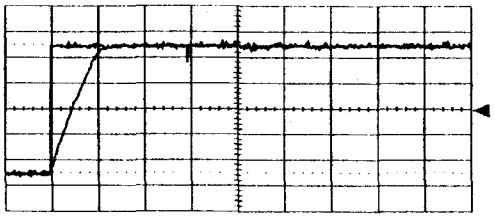
그림 9는 전체 하드웨어의 구성도를 나타낸다. 유도전동기 속도제어 시스템의 하드웨어는 고성능의 DSP320C31 제어



(X축: 0.5 sec/div, Y축: 2.5A/div)
(a) 토오크 성분 전류 지령치
(a) Reference current of torque component

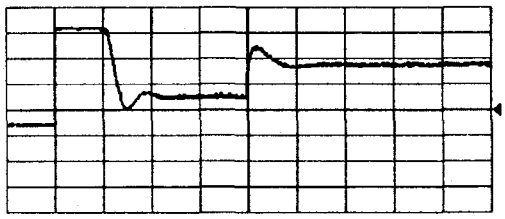


(X축: 0.5 sec/div, Y축: 2.5A/div)
(b) 토오크 성분 전류
(b) Real current of torque component

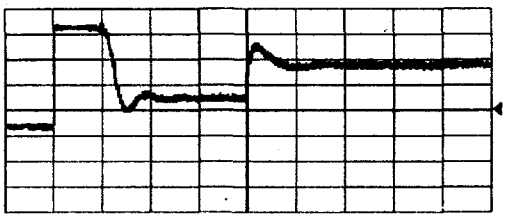


(X축: 0.5 sec/div, Y축: 400 rpm/div)
(c) 속도 응답
(c) Speed response for the reference change

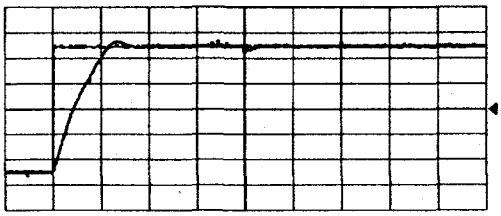
그림 10 PI 속도제어기와 PI 전류제어기의 응답
Fig. 10 Responses of PI speed controller and PI current controller



(X축: 0.5 sec/div, Y축: 2.5A/div)
(a) 토오크 성분 전류 지령치
(a) Reference current of torque component



(X축: 0.5 sec/div, Y축: 2.5A/div)
(b) 토오크 성분 전류
(b) Real current of torque component



(X축: 0.5 sec/div, Y축: 400 rpm/div)
(c) 속도 응답
(c) Speed response for the reference change

그림 11 퍼지 속도제어기와 퍼지 전류제어기의 응답
Fig. 11 Responses of fuzzy speed controller and fuzzy current controller

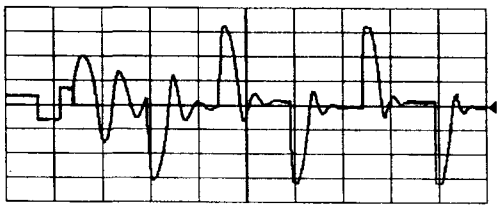
용 보드와 IGBT 전류제어형 PWM 인버어터로 구성된다. 모든 제어 알고리즘은 DSP에 의해 실시간으로 처리되며 제어용 프로그램은 C-언어로 작성되고, IBM-PC에서 Mountain-510 에뮬레이터를 통하여 DSP 타겟보드에 다운로드시켜 실시간 구현하였다. 전력회로의 인버터 스위칭 소자로는 FUJI사의 1500[V], 75[A]의 정격을 가지는 IPM을 사용하였다. 인버터의 암단락을 방지하기 위하여 데드타임은 IPM사양에 따라 5[㎲]로 설정했다.

각종 제어변수의 모니터링은 D/A를 통하여 오실로스코프로 관측하였다. 속도 샘플링 주기는 3[ms], 전류 샘플링 주기는 100[㎲]로 하였다. 부하 실험은 전동기 축에 직류타여자 발전기를 커플링하여 전등을 직·병렬하여 사용하였다. 그리고, 표 1에 시뮬레이션 및 실험에서 사용된 유도전동기 파라미터를 나타내었다. -1000~1000[rpm]으로 정역회전과 전등부하 인가 실험을 PI 속도제어기, 퍼지 속도제어기, 자기학습형 퍼지 속도제어기와 PI 전류제어기, 퍼지 전류제어기와 비교하여 실험하였다.

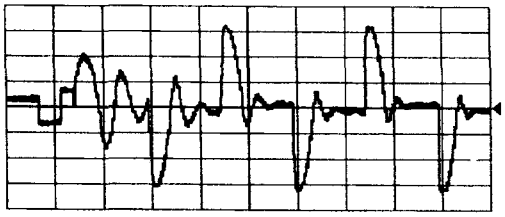
그림 11에는 퍼지 속도제어기와 퍼지 전류제어기를 사용했을 때의 토오크 성분 전류 지령치와 토오크 성분 전류, 그리

표 1 시뮬레이션과 실험에 사용된 전동기 상수 및 정격
Table 1 The Parameters of the induction motor used in the simulation study and the experimental work

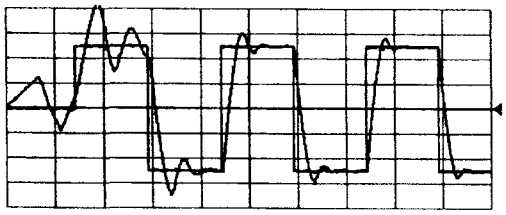
전압 : 220[V]	출력 : 2.2[kW]	극수 : 4[극]
회전수 : 1720[rpm]	주파수 : 60[Hz]	
Rs :0.687[Ω]	Ls : 0.08397[H]	
Rr :0.842[Ω]	Lr : 0.08528[H]	
Lm :0.08136[H]		
Bm :0.01[Kg · m2 /sec]	Jm :0.03 [Kg · m2]	



(X축: 0.5 sec/div, Y축: 2.5A/div)
(a) 토오크 성분 전류 지령치
(a) Reference current of torque component

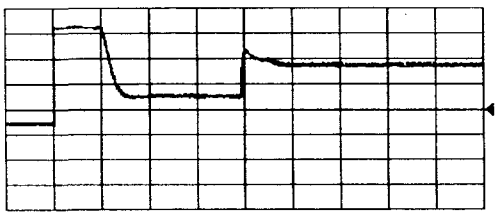


(X축: 0.5 sec/div, Y축: 2.5A/div)
(b) 토오크 성분 전류
(b) Real current of torque component

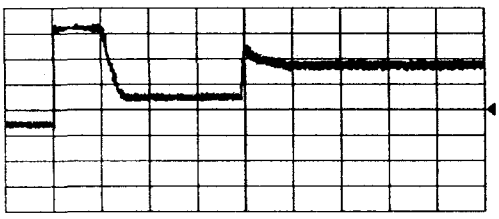


(X축: 0.5 sec/div, Y축: 400 rpm/div)
(c) 속도 응답
(c) Speed response for the reference change

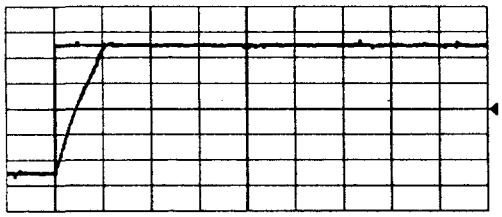
그림 12 자기학습형 퍼지제어기의 학습과정의 응답
Fig. 12 Responses of training of self-learning fuzzy controller



(X축: 0.5 sec/div, Y축: 2.5A/div)
(a) 토오크 성분 전류 지령치
(a) Reference current of torque component

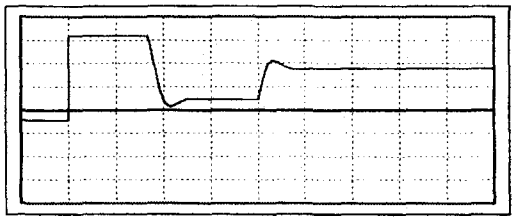


(X축: 0.5 sec/div, Y축: 2.5A/div)
(b) 토오크 성분 전류
(b) Real current of torque component

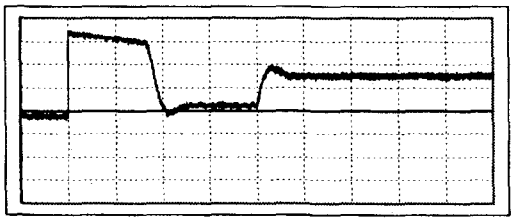


(X축: 0.5 sec/div, Y축: 400 rpm/div)
(c) 속도 응답
(c) Speed response for the reference change

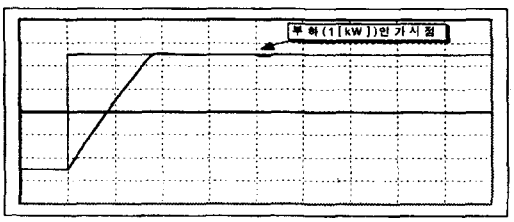
그림 13 자기학습형 퍼지 속도제어기와 퍼지 전류제어기의 응답
Fig. 13 Responses of self-learning fuzzy controller and fuzzy current controller



(X축: 0.5 sec/div, Y축: 2.5A/div)
(a) 토오크 성분 전류 지령치
(a) Reference current of torque component

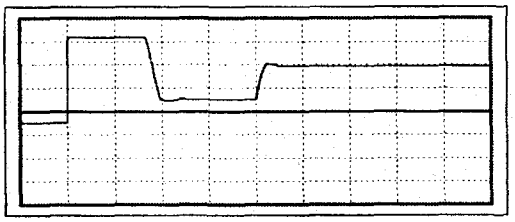


(X축: 0.5 sec/div, Y축: 2.5A/div)
(b) 토오크 성분 전류
(b) Real current of torque component

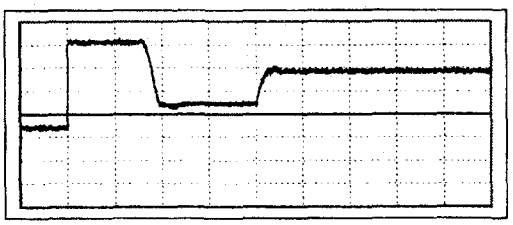


(X축: 0.5 sec/div, Y축: 400 rpm/div)
(c) 속도 응답
(c) Speed response for the reference change

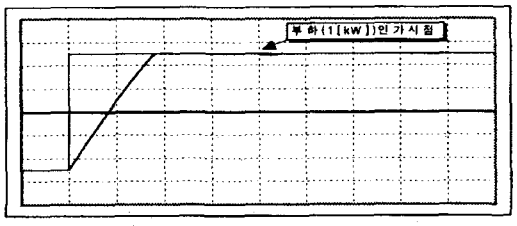
그림 14 PI 속도제어기와 PI 전류제어기의 응답 (J_m=0.05)
Fig. 14 Responses of PI speed controller and PI current controller (J_m=0.05)



(X축: 0.5 sec/div, Y축: 2.5A/div)
(a) 토오크 성분 전류 지령치
(a) Reference current of torque component



(X축: 0.5 sec/div, Y축: 2.5A/div)
(b) 토오크 성분 전류
(b) Real current of torque component



(X축: 0.5 sec/div, Y축: 400 rpm/div)
(c) 속도 응답
(c) Speed response for the reference change

그림 15 자기학습형 퍼지 속도제어기와 퍼지 전류제어기의 응답 (J_m=0.05)
Fig. 15 Responses of self-learning fuzzy controller and fuzzy current controller (J_m=0.05)

고 유도전동기의 실제속도와 기준속도의 파형을 보여주고 있는데, 그림 10의 PI 속도제어기와 PI 전류제어기를 사용했을 때보다 오버슈트와 진동은 비슷하며 응답시간과 부하에 대한 강인성면에서는 우수한 특성을 보였다.

그림 12는 무부하 상태에서 일정시간 동안 임의의 제어량을 내 보낸 후 -1000~1000[rpm]으로 변화하는 기준속도를 인가했을 때, 자기학습형 퍼지제어기는 신경망 에뮬레이터의 학습기능을 이용하여 gain tuning이 없는 상태에서도 빠른 시간내에 퍼지제어기의 소속함수와 제어규칙을 찾아가고 있음을 보여준다. 그림 13은 5회 학습 후의 자기학습형 퍼지 속도제어기와 퍼지 전류제어기를 사용한 것으로써 그림 11의 off-line으로 구성된 퍼지 속도제어기보다 정역운전에 대한 속도응답과 부하변화에 대하여 강인하며, 그림 10의 PI 전류제어기보다 응답시간과 부하변화에 대하여 우수한 특성을 보였다.

그리고, 그림 14, 15에서 제안한 자기학습형 퍼지 속도제어기는 스스로 최적의 제어이득을 가지도록 가중치와 소속함수의 파라미터를 변화시키는 적응능력을 가짐으로써 PI제어기에 비해서 전동기 관성변화 J_m=0.03에서 J_m=0.05로 강인한 제어성능을 보였다.

5. 결 론

본 논문에서는 최근 산업계에서 많은 관심의 대상이 되고 있는 지능제어기법에 의한 유도전동기 속도와 토오크 제어에 관하여 연구하였다. 정역운전과 부하에 대한 응답을 통하여 다음과 같은 결과를 얻을 수 있었다.

유도전동기 속도제어를 위하여 실시간 플랜트 동정이 가능

하도록 신경회로망 에뮬레이터를 이용하여 퍼지제어기의 소속함수와 제어규칙을 실시간으로 학습시키는 자기학습형 퍼지제어기를 제안하였다. 시뮬레이션과 실험을 통하여 제안된 자기학습형 퍼지제어기는 5회 정도의 예비학습만으로도 기준속도를 추종하는 성능이 우수함을 알 수 있었고, 갑작스런 부하의 변동에 대하여 강인성을 가짐을 확인할 수 있었다.

토오크 성분과 자속 성분의 전류제어는 look-up table을 이용한 일반적인 퍼지제어기를 사용하였다. 그 결과 오버슈트와 진동은 PI제어기와 비교하면 오버슈트와 진동은 비슷하나 응답시간과 부하변화에 대한 강인성면에서는 우수한 특성을 보였다.

참 고 문 헌

[1] 고종선, 이정훈, 윤명중, "훠지 로직을 이용한 브러쉬 없는 직류전동기의 파라미터의 변화와 외란에 둔감한 위치제어," 대한전기학회 논문지, vol.40, No.10, pp.1037~1048. 1991.

[2] I. Miki, N. Nagai, S. Nishiyama and T. Yamada, "Vector control of induction motor with fuzzy PI controller," IEEE IAS Conf. Rec., pp.341~346, 1991.

[3] Jyh-Shing R. Jang, "Self-Learning Fuzzy Controllers Based on Temporal Back Propagation", IEEE Trans. on Neural Networks, vol.3, No.5, pp.714~723, Sep. 1992.

[4] Marzuki Khalid and Rubiyah Yusof, "Neuro-Control and its Application", Springer, 1996.

[5] Bimal. K. Bose, "Power Electronics and Variable Frequency Drives", IEEE Press, 1996.

[6] 김세찬, 원충연, "신경회로망을 이용한 유도전동기 속도 제어," 대한전기학회 논문지, vol.45, No.1, pp.42~53, 1996.

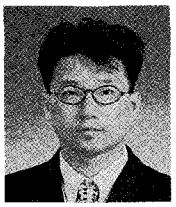
[7] 김윤호, 이병순, 성세진, "초고속 유도전동기 구동을 위한 신경회로망 제어기 설계", 전력전자학회 논문지 제2권, 제1호, pp.39~45, 1997. 3.

[8] 박영민, 김연충, 김재문, 원충연, 김영렬, 김학성, "자기학습형 퍼지제어기에 의한 유도전동기 고성능 속도제어 에 관한 연구", 대한전기학회 추계학술대회 논문집, pp.505~508, 1997.

[9] 김준석, "공간 전압 벡터 PWM의 새로운 기법", 대한전기학회 논문지, vol.44, No.7, pp.865~874, 1995.

[10] H. W. Van der Broeck, H. C. Skudelny, "Analysis and Realization of a Pulse Width Modulator Based on Voltage Space Vector", IEEE Trans. on Ind. Applicat. vol. IA-24, pp.142~150, 1988.

〈저 자 소 개〉



박영민(朴英珉)

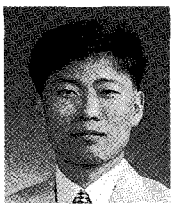
1969년 12월 8일생. 1996년 성균관대 공대, 전기공학과 졸업. 1998년 동 대학원 전기공학과 졸업(석사). 현재 현대중공업 마북리연구소 전력전자연구실 연구원.

김덕헌(金德憲)

1964년 8월 15일생. 1990년 성균관대 공대 전기공학과 졸업. 1992년 동 대학원 전기공학과 졸업(석사). 1995년 동 대학원 전기공학과 박사과정 수료. 현재 가톨릭 상지대학 전기과 조교수.

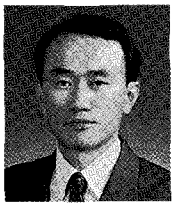
김연충(金淵忠)

1970년 2월 14일생. 1995년 성균관대 공대 전기공학과 졸업. 1997년 동 대학원 전기공학과 졸업(석사). 현재 동 대학원 전기전자 및 컴퓨터공학부 박사과정.



김재문(金才文)

1967년 9월 6일생.1994년 성균관대 공대 전기공학과 졸업.1996년 동 대학원 전기공학과 졸업(석사). 현재 동 대학원 전기공학과 박사과정.



원충연(元忠淵)

1978년 성균관대 공대 전기공학과 졸업. 1980년 서울대 공대 대학원 전기공학과 졸업(석사) 1987년 동 대학원 전기공학과 졸업(공박). 1991년~1992년 미국 테네시 주립대학 전기공학과 객원교수. 현재 성균관대 전기전자 및 컴퓨터공학부 교수, 당 학회 학술이사.