한국 퍼지 및 지능시스템학회 논문지 1999, Vol. 9, No. 5 (1999)463~471

뉴럴 네트워크 방식의 벡터제어에 의한 유도전동기의 속도 제어

The Speed Control of Vector controlled Induction Motor Based on Neural Networks

이동빈 · 유창완 · 홍대승 · 임화영

Dong-Bin Lee, Chang-Wan Ryu, Dae-Seung Hong and Wha-Yeong Yim

광운대학교 제어계측공학과

ABSTRACT

This paper presents a vector controlled induction motor is implemented by neural networks system compared with PI controller for the speed control. The design employed the training strategy with Neural Network Controller(NNC) and Neural Network Emulator(NNE) for speed. In order to update the weights of the controller, First of all, Emulator updates its parameters by identifying the motor input and output. next, it supplies the error path to the output stage of the controller using backpropagation algorithm. As Controller produces an adequate output to the system due to neural networks learning capability, Vector controlled induction motor speed follows the reference speed. The simulation result demonstrated that In spite of adding the mechanical load changing the motor characteristics, actual motor speed with based on neural network system follows the reference speed better than that of linear PI speed controller.

1. 서 론

유도 전동기는 견고하고 고속이 가능하며 경제성등이 우수하나 가변속 제어가 어려워 정속도용으로 주로 사용되어 왔다. 반면에 직류 전동기는 토오크와 자속을 독립적으로 제어하여 속도 제어가 쉬운 장점이 있지만 정류자와 브러쉬에 의해 속도 대 power의 근본적인 제한과 유지 보수등이 유도 전동기에 비해 제약이 따른다. 유도 전동기를 직류 전동기처럼 속도 제어하기 위해 자속 기준 제어나 벡터 제어등 많은 적응 제어 기법등이 연구되고 있으나 복잡한 알고리즘을 필요로 한다. 속도 제어기로서 선형 PI 또는 PID 이론을 적용하더라도 최적의 파라미터를 선정하기에 어려움이 존재한다. 또한 속도의 변환, 부하시에는 적절한 조정을 필요로 한다. 최근에는 지능 제어 기법으로서 퍼지나 뉴럴, 유전자 알고리즘등을 이용한 비선형 제어[1,2] 또는 선형 PI, PID와 결합된 형태로 제어하는 연구가 진행되고 있다[3,4]. 뉴럴 네트워크를 이용한 대부분의 제어기 경우, 학습에 필요한 입출력해를 구하기가 어렵다. 이런 경우 플랜트의 입,출력 구조와 동일한 뉴럴 네트워크의 입·출력구조를 선택하여 시스템의 모델링을 근사화하는 방법이 연구되고 있다[1-5].

본 논문에서는 뉴럴 네트워크 시스템을 이용하여 유도 전동기의 속도를 제어하는 기법을 적용하였다. 뉴럴 네트워크 시스템은 제어기와 에뮬레이터로 구성하였고 에뮬레이터는 플랜트의 입·출력구조와 동일한 구조로서, PWM 인버터, 유도 전동기등을 포함한 전체 벡터 시스템을 근사화한다. 뉴럴 네트워크 시스템의 학습은 유도전동기의 실제속도와 기준속도와의 오차를 이용하여 오차 역전파 학습 알고리즘을 적용하였다. 학습시 가중치 조절은 전체 오차의 합이 최소가 되도록 최급 강하법(Steepest Decent)을 이용하였다. 무부하시의 특성을 포함하여 유도 전동기에 부하를 인가하여 시스템의 특성이 변하는 환경에서, 적응 뉴럴 네트워크 시스템과 선형 PI 제어기와의 기준 속도의 추종성을 비교함으로서 뉴럴 네트워크의 학습 능력과 적응성을 확인하였다.

2. 본 론

2.1. 뉴럴 네트워크를 이용한 속도 제어

그림 1은 실제 속도 y(i)가 기준 속도 x(i)를 추종하도록 뉴럴 네트워크를 이용하여 속도 제어를 하는 벡터 제어 유도 전동기 시스템이다.

에뮬레이터는 플랜트의 입, 출력 쌍을 학습하여 벡터 제어 유도 전동기를 근사화한다. 제어기는 근사화된 에뮬레이터를 이용하여 최적의 입력값 u(i)를 출력

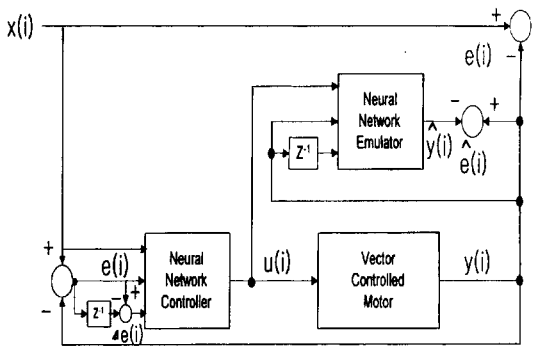


그림 1. 뉴럴 네트워크를 이용한 유도 전동기 속도 제어 시스템

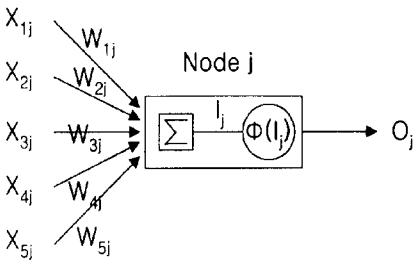


그림 2. 역전파 퍼셉트론의 뉴런 j
(Node j of a backpropagation MLP)

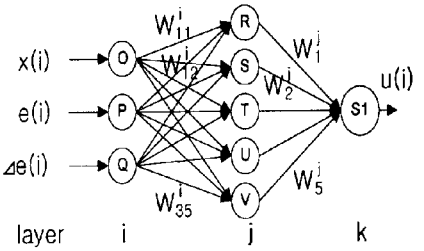


그림 3. 뉴런 네트워크 제어기
(Neural Network controller)

함으로서 목표 출력이 나오도록 한다. 뉴럴 네트워크를 이용한 속도 제어 시스템의 구성 요소로는 벡터 제어 유도 전동기 시스템과 피드포워드(feedforward) 뉴럴 네트워크 제어기 및 에뮬레이터로 구성하였다. 뉴럴 네트워크의 각 노드의 활성함수는 ±1을 갖고 있는 hypertangent 함수 $\Phi(I_j)$를 적용하였다. 에뮬레이터 은닉층의 j번째 노드(뉴런)을 그림 2에 표현하였다.

여기서 x_{ij}는 입력, W_{ij}는 i, j 노드간의 가중치, I_j는 가중치된 입력들의 합이고 O_j는 활성함수의 출력이다.

그림 3은 뉴럴 네트워크로 구성된 속도 제어기이다. 제어기는 3개의 층으로 구성하였고 입력층 I, 은닉층 j 그리고 출력층 k이다. 여기서, 입력층의 뉴런은 유도 전동기의 상태 변수를 입력화하였다. 입력 변수로서 x(i)는 기준속도, e(i)는 기준속도 x(i)와 실제 속도 y(i)와의 차이고 Δe는 그 오차의 변화량이다.

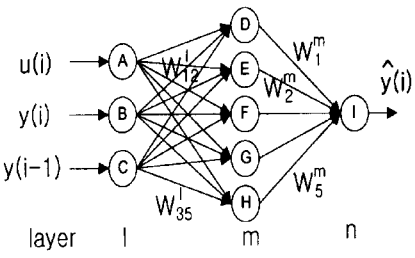


그림 4. 뉴럴 네트워크 에뮬레이터
(Neural Network Emulator)

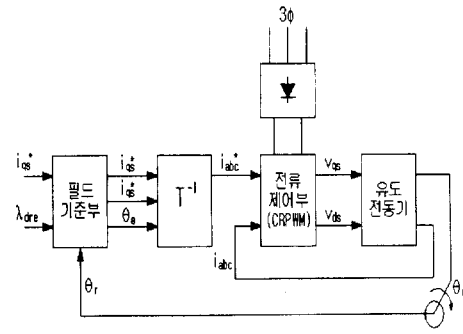


그림 5. 유도 전동기의 벡터 시스템

은닉층은 5개 뉴런으로 구성되었고 은닉층의 가중치 W_{ij}^{i}는 15개이고 출력층의 가중치 W_{jk}^{j}는 5개로서 총 20개이다. 출력층의 출력은 플랜트 및 에뮬레이터에 가해지는 u(i)이다.

그림 4는 유도 전동기 벡터 제어 시스템을 근사화하는 에뮬레이터이다.

에뮬레이터는 3개의 층으로 구성하였고 각각 입력층 l, 은닉층 m 그리고 출력층 n이다. 입력층은 플랜트와 동일한 u(i), 유도 전동기의 출력 y(i)와 그 전상태인 y(i-1)을 입력 변수로 하였다.

은닉층의 뉴런은 5개이면서 은닉층의 가중치 W_{lm}^{l}는 15개이고 출력층의 가중치 W_{mn}^{m} 은 5개로서 총 20개이다. $\hat{y}$은 출력층의 뉴런으로 에뮬레이터의 속도이다.

2.2 유도 전동기의 벡터 시스템

플랜트인 유도 전동기의 벡터 시스템은 그림 5와 같이 크게 3개의 부분으로 구성되었다. 첫째는 필드 기준(Field Orientation)부, 두번째는 전류 제어 PWM 인버터부 그리고 유도 전동기의 기계적인 모델링부이다.

유도 전동기는 변수 및 파라미터들이 상호 결합된

다변수 비선형 시스템으로서 복잡한 상호 간섭과 과도적인 응답 특성을 나타낸다. 그러므로 독립적인 자속, 토오크 특성을 가지고 제어하는 직류 전동기보다 복잡한 제어기 구성을 갖는다. 따라서 그 등가 모델은 과도 상태의 동특성을 잘 나타내어야 한다. 플랜트인 벡터 제어 시스템은 자속 기준 원리(flux orientation principle)를 이용하여 고정자 입력 전류를 자속 성분으로 분리한 d축과 토오크 성분으로 분리한 q축으로 분리할 수 있다.

유도 전동기의 전압식과 자속식은 고정자와 회전자에 대해 dq축 해석을 하고 그 등가 모델을 그림 6에 나타내었다.

식(1)은 dq축 고정자 전압식으로서 고정자 전류 i_{qds}와 고정자 자속 λ_{qds}만으로 표현하였다.

$$\begin{aligned} v_{qs} &= r_s i_{qs} + p\lambda_{qs} + w_e\lambda_{ds} \\ v_{ds} &= r_s i_{ds} + p\lambda_{ds} - w_e\lambda_{qs} \end{aligned} \tag{1}$$

여기서, ω_e는 동기각속도, r_s는 고정자 저항이고 p는 미분 연산자 d/dt이다.

식(2)는 고정자의 dq축 쇄교 자속식을 나타내었다.

$$\begin{aligned} \lambda_{qs} &= L_s i_{qs} + L_m i_{qr} \\ \lambda_{ds} &= L_s i_{ds} + L_m i_{dr} \end{aligned} \tag{2}$$

여기서, L_s는 고정자 인덕턴스, L_m은 고정자의 자화 인덕턴스이다.

식(3)은 회전자 전압식을 나타냈고 농형 유도 전동기이므로 이상적인 변압기의 원리에 의해 0이다.

$$\begin{aligned} 0 &= r_r i_{qr} + p\lambda_{qr} + (w_e - w_r)\lambda_{dr} \\ 0 &= r_r i_{dr} + p\lambda_{dr} - (w_e - w_r)\lambda_{qr} \end{aligned} \tag{3}$$

여기서 r_r은 회전자 저항, ω_r은 회전자의 각속도이다.

식(4)는 회전자의 dq축 쇄교 자속식을 나타내었다.

$$\lambda_{qr} = L_r i_{qr} + L_m i_{qs}$$

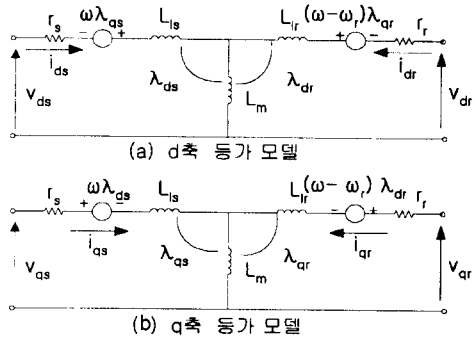


그림 6. 유도 전동기의 등가모델

$$\lambda_{dr} = L_r i_{dr} + L_m i_{ds} \tag{4}$$

여기서 L_r은 회전자의 인덕턴스이다.

식(1)~(4)는 동기축을 따라서 회전하는 좌표계를 사용하였고 고정축 좌표계 사용시에 동기 각속도 ω_e는 0가 된다.

그림 7은 d축과 q축 그리고 회전자의 토오크 및 속도를 발생시키는 기계적인 모델링부로 구성하였다.

식(1)의 dq축 고정자 전압식으로부터 고정자 자속식을 식(5)와 같이 구할 수가 있다.

$$\begin{aligned} \lambda_{qs} &= \int (v_{qs} - r_s i_{qs})dt \\ \lambda_{ds} &= \int (v_{ds} - r_s i_{ds})dt \end{aligned} \tag{5}$$

식(6)의 dq축 고정자 전류는 고정자 자속식(5)와 회전자 자속식 (9)를 이용하여 구할 수 있다.

$$\begin{aligned} i_{qs} &= \frac{1}{\sigma L s}\left(\lambda_{qs} - \frac{L_m}{L_r}\lambda_{qr}\right) \\ i_{ds} &= \frac{1}{\sigma L s}\left(\lambda_{ds} - \frac{L_m}{L_r}\lambda_{dr}\right) \end{aligned} \tag{6}$$

여기서 σL_s는 고정자 천이 누설 인덕턴스(stator transient inductance)로서 식(7)과 같다.

$$\sigma L_s = L_s(1 - L_m^2 / (L_s L_r)) \tag{7}$$

식(8)의 dq축 회전자 전류도 식(5)와 식(9)를 이용하여 구할 수 있다.

$$\begin{aligned} i_{qr} &= \frac{1}{\sigma L s}\left(\lambda_{qr} - \frac{L_m}{L_s}\lambda_{qs}\right) \\ i_{dr} &= \frac{1}{\sigma L s}\left(\lambda_{dr} - \frac{L_m}{L_s}\lambda_{ds}\right) \end{aligned} \tag{8}$$

dq축 회전자 자속 λ_{dqr}은 dq축 회전자 전압식 (3)으로부터 식(9)와 같이 구할 수 있다.

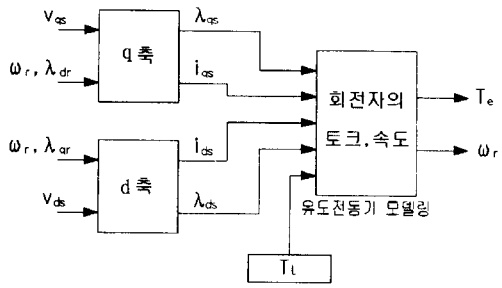


그림 7. 유도 전동기의 모델링부

$$\lambda_{qr} = \int(\omega_r\lambda_{dr} - r_r i_{qr})dt$$
$$\lambda_{dr} = -\int(\omega_r\lambda_{qr} + r_r i_{dr})dt \quad (9)$$

그림 7의 유도 전동기의 기계적인 모델링의 해인 식(10)의 좌변 T_e는 토오크식으로서 식(11)과 같이 나타낼 수 있다.

$$T_e = J_m\frac{d}{dt}\omega_r + B_m\omega_r + T_L \quad (10)$$

$$T_e = \left(\frac{3P}{4}\right)(\lambda_{qs}i_{ds} - \lambda_{ds}i_{qs}) \quad (11)$$

여기서 J_m은 관성 모멘트, B_m은 모터의 베어링 마찰 계수 그리고 P는 모터의 극 수이다. 식(10)의 우변에서 T_L은 기계적인 부하로서 무부하일 때에는 0을, 부하시에는 정격 토오크 만큼의 부하를 인가하였고 B_m은 0으로 하였다. 따라서 속도 ω_r은 토오크식을 적분함으로서 해를 구할 수가 있다.

유도 전동기 벡터 시스템에서 필드 기준부는 q축 회전자 자속 λ_{qr}을 0으로 만드는데 목적이 있고 이를 유지하기 위해 식 (12)와 같이 회전자 자속각 θ_e를 출력한다.

$$\theta_e = \int_0^t(\omega_{slip} + \omega_r)dt \quad (12)$$

식 (12)는 직류 전동기의 브러쉬 역할과 같은 것으로서 유도 전동기의 회전자 각도 θ_r과 dq 전류 명령치로부터 식(13)과 같이 계산된 θ_{slip}을 합한 것이다.

슬립 각속도 ω_{slip}은 고정자축과 회전자축간의 상대 속도이다.

$$\omega_{slip} = \frac{1}{\tau_r}\frac{i_{qs}}{i_{ds}} \quad (13)$$

여기서 $\tau_r = L_r/r_r$는 회전자 시정수이다.

회전자 자속각 θ_e는 유도 전동기가 직류 전동기처럼 자속과 토오크를 독립적으로 작용하기 위해 dq 전류에 자속각을 순시적으로 보정하여 동기화 하여 주고 있다. 간접 벡터 방식은 슬립식 (13)에 기반을 두고 있으며 필드 기준을 하기 위한 필요 충분 조건이다. 슬립 관계가 만족된다면 i_{ds}는 회전자 자속과 동일선상이 된다. d축 전류 명령치는 회전자 전압식(3)과 회전자 자속식(4)를 이용하여 식(14)와 같고 회전자 자속에 일차 지연이고 회전자 시정수 τ_r만큼 늦다.

$$i_{ds} = (\lambda_{dr} + \tau_r p\lambda_{dr})/L_m \quad (14)$$

동기화된 dq전류 지령치는 고정축으로 축변환을 한다. 동기 회전 좌표계에서 고정축 좌표로의 변환은 식 (15)와 같이 축변환식을 이용한다.

$$i_{qds} = [e^{j\theta}]i^{e}_{qds}$$

$$\begin{bmatrix} i_{qs} \\ i_{ds} \end{bmatrix} = \begin{bmatrix} \cos\theta_e & \sin\theta_e \\ -\sin\theta_e & \cos\theta_e \end{bmatrix}\begin{bmatrix} i^e_{qs} \\ i^e_{ds} \end{bmatrix} \quad (15)$$

고정축 기준 값으로 변환된 뒤 3상 좌표 변환을 하고 이는 전류 제어 PWM 인버터부의 3상 전류 지령치가 된다. 그림 5의 T^{-1}부이다.

2상에서 3상으로의 변환은 좌표 변환 행렬식 (16)을 이용하여 구할 수 있다.

$$i^*_{abcs} = [T^s_{qd0}]^{-1}i_{qds}$$

$$[T^s_{qd0}]^{-1} = \begin{bmatrix} 1 & 0 & 1 \\ -\frac{1}{2} & -\frac{\sqrt{3}}{2} & 1 \\ -\frac{1}{2} & \frac{\sqrt{3}}{2} & 1 \end{bmatrix} \quad (16)$$

전류 제어 PWM 인버터부는 비교적 오차가 작으며 실현이 용이한 히스테리시스 전류 제어기를 적용하였다.실제 전류 i_{abc}가 기준 전류 i^*_{abc}를 추종할 수 있도록 순시 전류를 제어하고 히스테리시스 폭에 의해 PWM 전압 파형을 출력한다. 실제의 상전류 i_{abc}는 유도 전동기의 출력 전류 i_{qds}를 상변환식 (16)을 이용하여 3상 변환한다. 전류 제어 PWM 인버터의 3상 출력 전압 v_{abc}는 식 (17)과 같이 2상 전압으로 dq 변환하고 유도 전동기의 입력이 된다.

$$v_{qds} = [T^s_{qd0}]\ v_{abcs} \quad (17)$$

2.3 선형 PI 속도 제어기의 설계

뉴럴 네트워크와 성능을 비교하기 위하여 선형 PI 속도 제어기를 그림 8과 같이 설계하였다.

속도 명령 대 속도 응답에 대한 시스템의 전달 함수를 구하면 식(18)과 같다.

$$M(s) = \frac{\omega_r}{\omega_r^*} = \frac{\frac{K_p}{J_m}s + \frac{K_i}{J_m}}{s^2 + \frac{(K_p + B_m)s}{J_m} + \frac{K_i}{J_m}} \quad (18)$$

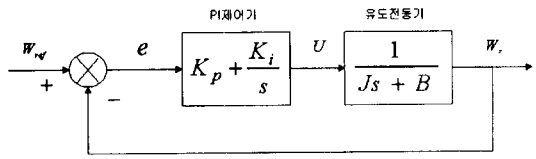


그림 8. 선형 PI 제어시스템의 전달함수 블록도

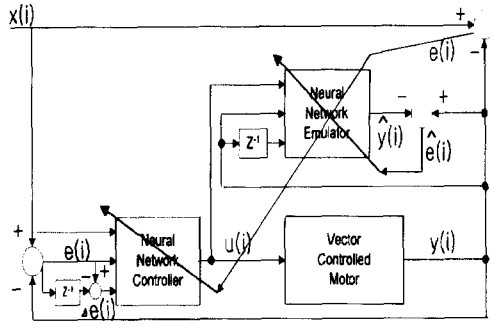


그림 9. 뉴럴 네트워크를 이용한 유도 전동기 속도 제어 시스템

이는 2차계 시스템의 전형적인 전달 함수의 분모와 같으므로 여기서 $\xi = 0.707$, $\omega_n = 15$일 때 $K_p = 0.42$, $K_i = 4.5$를 구하였다.

2.4 뉴럴 네트워크 속도 제어시스템의 학습 알고리즘

그림 9에는 뉴럴네트워크를 이용한 속도 제어 시스템의 학습 알고리즘을 나타내었다.

기준 속도 x는 제어기에 입력되어 임의의 u값을 출력한다. 이 u값은 유도 전동기의 벡터 시스템의 입력으로서 q축 고정자 전류 i_{qs}값이 되고 일정 자속에 의한 d축 전류 i_{ds}값과 함께 벡터 시스템이 동작된다. 이 때 플랜트와 에뮬레이터의 출력은 각각 y와 $\hat{y}$이다. 에뮬레이터의 가중치를 조정하기 위해서는 이 둘의 오차 $\hat{e}$를 역전파시켜 사용한다.

제어기의 가중치 학습은 기준 속도 x와 실제 속도 y와의 차인 e를 이용한다.

이 때 제어기의 가중치를 학습시키기 위해 오차 e를 에뮬레이터를 통해 역전파(backpropagation)시켜서 가중치를 정정하게 되므로 에뮬레이터의 성능이 본 제어기의 성능을 좌우할 수 있다.

즉, 에뮬레이터의 은닉층의 오차의 합을 제어기 출력단에 역전파시켜 제어기의 출력단 및 은닉층의 오차항을 계산하고 제어기 가중치를 조정하여 $y = x$가 되도록 학습시킨다. 반복적으로 역전파 학습 알고리즘을 이용하여 최적의 입력값 u(i)를 출력함으로서 유도 전동기의 실제 속도가 기준 속도와 일치하도록 한다. 그림 10은 본 논문의 제어기 알고리즘의 흐름도를 나타내었다. 평가 함수가 최소가 되는 조건으로 제어기 노드 i, j간의 가중치 W_{ij}의 조정은 식(19)와 같이 구하였다.

$$\frac{\partial E}{\partial w_{ij}} = 0 \tag{19}$$

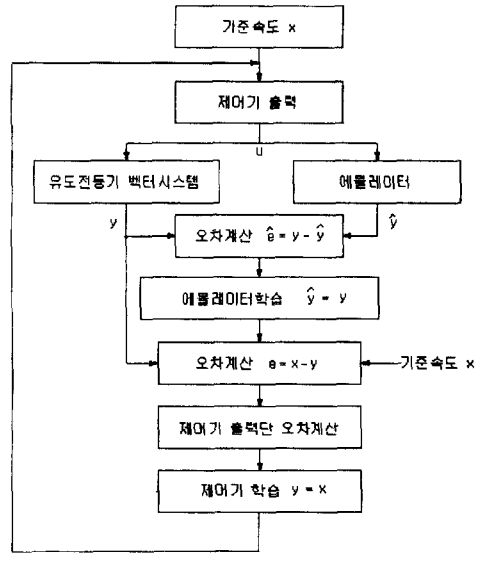


그림 10. 학습 흐름도

식 (19)는 에러의 변화에 대해 가중치의 변화량이 0이 되는 조건이다. 최급 강하법을 써서 가중치 변분을 에러가 감소하는 방향으로 계산하였다. 여기서 η는 학습율이다.

$$\Delta W_{i,j} = -\eta \frac{\partial E}{\partial W_{i,j}} \qquad \eta > 0 \tag{20}$$

지역 최저치(local minimum)에 빠지는 것을 방지하기 위해 식(21)과 같이 모멘텀을 적용한다.

$$\Delta W(k+1) = \alpha \Delta W(k+1) + (1-\alpha)\Delta W(k),\ 0 \le \alpha \le 1 \tag{21}$$

여기서 α는 모멘텀 계수로서 0.9를 선정하였다.

따라서 일반화된 가중치 조절식은 식 (22)와 같다.

$$W(k+1) = W(k) + \Delta W(k+1) \tag{22}$$

2.4.1 에뮬레이터의 학습

뉴럴 네트워크 에뮬레이터의 학습은 유도 전동기의 실제 출력 y와 에뮬레이터의 출력 $\hat{y}$과의 오차를 이용한 오차 함수 $\hat{E}$을 식(23),(24)와 같이 구할 수 있다.

$$\hat{e}(k) = (y_k - \hat{y}_k) \tag{23}$$

$$\hat{E} = \frac{1}{2}\sum_{k=1}^{P} \hat{e}_k^2 \tag{24}$$

제어기 출력단의 오차항을 이용하여 식 (25)와 같이 m층의 가중치를 조정할 수 있다.

$$\Delta W_m = -\eta_e \frac{\partial \hat{E}}{\partial W_m} \tag{25}$$

여기서 η_e는 에뮬레이터의 학습율로서 0.22를 선정하였다. 우변은 일반화된 델타 규칙과 체인 룰을 적용하면 식(26)이다.

$$\begin{aligned} -\eta_e \frac{\partial \hat{E}}{\partial W_m} &= -\eta_e \frac{\partial \hat{E}}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial O_n} \frac{\partial I_n}{\partial I_n} \frac{\partial I_n}{\partial W_m} \\ &= -\delta_n \frac{\partial I_n}{\partial W_m} \end{aligned} \tag{26}$$

여기서 δ_n은 n번 째 layer, 즉 출력층의 오차항이다. 결과로 출력되는 함수는 식(27)과 같다.

$$-\eta_e \frac{\partial \hat{E}}{\partial W_m} = \eta_e \sum_{k=1}^{P} \hat{e}_k (1-\tanh^2(I_n)) O_m \tag{27}$$

에뮬레이터 은닉층에서의 오차항 δ_m은 에뮬레이터 출력층의 오차를 이용하여 구할 수 있고 이 때 l층의 가중치 조절은 식 (28)과 같다.

$$-\eta_e \frac{\partial \hat{E}}{\partial W_l} = -\delta_n \frac{\partial I_n}{\partial O_m} \frac{\partial O_m}{\partial I_m} \frac{\partial I_m}{\partial W_l} = -\delta_m O_l \tag{28}$$

2.4.2 제어기의 학습

뉴럴 네트워크 제어기의 가중치를 변화하기 위해서는 먼저 에뮬레이터가 정확히 실제 플랜트를 근사화해야 한다. 에뮬레이터가 플랜트로서 근사화한 후 제어기를 학습한다. 제어기가 한 번 학습될 때마다 모터의 입·출력 쌍이 변하므로 에뮬레이터가 플랜트와 동일하도록 재 학습을 시킨다. 목표 속도 x와 유도 전동기의 실제 출력 y와의 오차식 (29)로부터 전체 자승의 합인 식 (30)을 평가 함수로서 사용하였다.

$$e(k) = (x_k - y_k) \tag{29}$$

$$E = \frac{1}{2} \sum_{k=1}^{P} e_k^2 \tag{30}$$

에뮬레이터를 학습시킬 때 에뮬레이터의 가중치 조절은 식(31)이다.

$$\Delta W_m = -\eta_c \frac{\partial E}{\partial W_m} \tag{31}$$

여기서 η_c는 제어기의 학습율로서 0.4를 선정하였다. 에뮬레이터 출력층의 오차항 계산은 식 (31)의 우변으로부터 식(32)와 같이 구한다.

$$\begin{aligned} -\eta_e \frac{\partial E}{\partial W_m} &= -\delta_n \frac{\partial I_m}{\partial W_m} \\ &= -\eta_c \frac{\partial E}{\partial y} \frac{\partial y}{\partial O_n} \frac{\partial O_n}{\partial I_n} \frac{\partial I_n}{\partial W_m} \\ &= -\eta_c \sum_{k=1}^{P} e_k(-1)(1-\tanh^2(I_n)) O_m \end{aligned} \tag{32}$$

여기서 δ_n은 n번 째 layer, 즉 출력층의 오차항이다. 에뮬레이터 은닉층에서의 오차항은 출력층의 오차를 이용하여 구할 수 있고

$$\begin{aligned} -\eta_c \frac{\partial E}{\partial W_l} &= -\delta_m O_l \\ &= -\delta_n \frac{\partial I_n}{\partial O_m} \frac{\partial O_m}{\partial I_m} \frac{\partial I_m}{\partial W_l} \\ &= -\delta_n W_m (1-\tanh^2(I_l)) O_l \end{aligned} \tag{33}$$

제어기의 가중치를 조정하기 위해서 먼저 에뮬레이터 입력층과 제어기의 출력층에서의 오차항을 구하면, 에뮬레이터의 오차항이 역전파되므로 은닉층의 각 노드의 오차항의 합이 된다.

$$\delta_l = \sum \delta_m = \delta_{m1} + \delta_{m2} + \delta_{m3} + \delta_{m4} + \delta_{m5} \tag{34}$$

따라서 제어기 출력층에서의 가중치 조절은

$$\Delta W_j = -\eta_c \frac{\partial E_u}{\partial W_j} \tag{35}$$

여기서 우변은 마찬가지로 델타 규칙과 체인 룰로서 식(36)과 같이 구한다.

$$\begin{aligned} -\eta_c \frac{\partial E_u}{\partial W_j} &= -\delta_l \frac{\partial O_k}{\partial I_k} \frac{\partial I_k}{\partial W_j} \\ &= -\delta_k \frac{\partial I_k}{\partial W_j} \\ &= -\delta_l (1-\tanh^2(I_k)) O_j \end{aligned} \tag{36}$$

마지막으로 제어기 은닉층에서의 오차항은 출력층의 오차를 이용하여 구할 수 있다.

$$\begin{aligned} -\eta_c \frac{\partial E_u}{\partial W_i} &= -\delta_l \frac{\partial O_k}{\partial I_k} \frac{\partial I_k}{\partial O_k} \frac{\partial O_j}{\partial I_j} \frac{\partial I_j}{\partial W_i} \\ &= -\delta_j W_j (1-\tanh^2(I_j)) O_i \end{aligned} \tag{37}$$

가중치 조절은 식(22)를 이용한다.

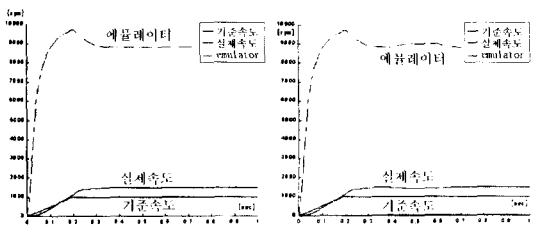
2.5 시뮬레이션 결과

표 1에 본 논문의 시뮬레이션에 사용된 유도 전동기의 파라미터를 나타내었다.

출력은 5hp으로서 유도 전동기의 정격 회전수는 슬

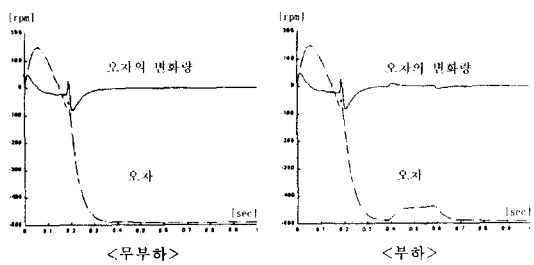
표 1. 유도 전동기 파라미터

모터 정격	5hp	$r_s(\Omega)$	0.2
정격 전압	220V	$r_r(\Omega)$	0.18
극 수	4p	L_s(mH)	180
정격 전류	13.5	L_r(mH)	180
회 전 수	1748rpm	L_m(mH)	176
주 파 수	60Hz	J_m(kgm^2)	0.02



<무부하> <부하>

그림 11. 초기 상태의 기준, 실제, 에뮬레이터 속도



<무부하> <부하>

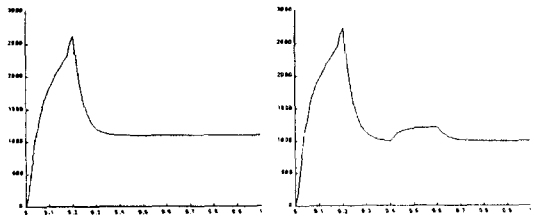
그림 12. 기준속도와 실제속도간의 오차와 오차변화량

립(s) 0.0248을 감안하면 1748 rpm이고 관성 J_m은 0.02 kgm^{2}이다. 그림의 좌측은 무부하시를, 우측은 부하시이다. 부하는 정격 토오크에 해당하는 힘을 가해 주었다. 그림 11은 기준 속도 1000 rpm이 제어기에 입력되었을 때 임의의 u값이 각 플랜트 및 에뮬레이터가 작동되었을 경우이다. 학습 되지 않은 임의의 상태이고 기준, 실제 그리고 에뮬레이터의 속도를 나타내었다. 실제 속도는 약 1500 rpm이고 에뮬레터는 학습되지 않은 상태로서 약 9000 rpm으로 큰 값을 가지고 있음을 볼 수 있다.

그림 12는 기준속도 x와 실제 속도 y간의 오차와 그 오차의 변화량 Δe를 나타내었다.

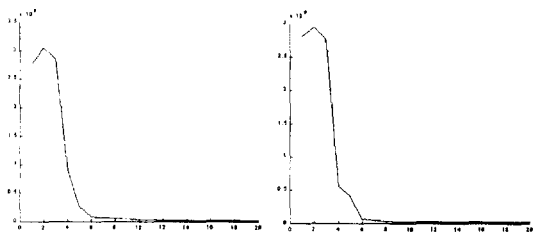
초기의 에뮬레이터를 유도 전동기 시스템과 동일하게 근사화 하기위해 20회 학습시 에뮬레이터의 출력이 그림 13에 나타나있다.

에뮬레이터의 속도 $\hat{y}$와 실제 속도 y와의 차를 이용한 전체 오차 자승의 합의 변화량 $\hat{E}$이 그림 14에 나타나 있다. 약 12회 정도 학습하면 오차가 최소에 이



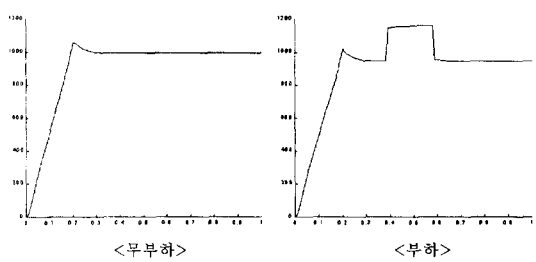
<무부하> <부하>

그림 13. 에뮬레이터 출력



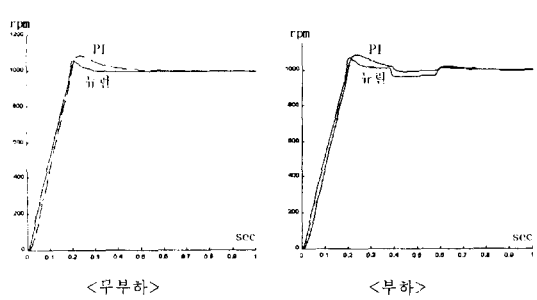
<무부하> <부하>

그림 14. 에뮬레이터 학습시 전체 오차의 변화



<무부하> <부하>

그림 15. 에뮬레이터 출력



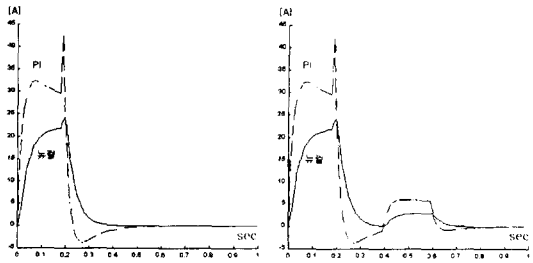
<무부하> <부하>

그림 16. 학습을 한 후 실제 속도

른다.

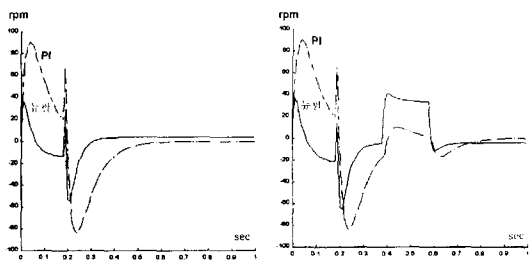
제어기를 학습하기 전에 에뮬레이터를 기준 속도와 실제 속도와의 차인 e로부터 오차함수 E를 이용하여 에뮬레이터를 학습시킨 결과가 그림 15에 나타나있다.

이 때에도 에뮬레이터는 20회를 학습하였다. 부하가 발생시 오버슈트가 존재하지만 에뮬레이터가 기준



<무부하> <부하>

그림 17. 제어기 출력 u(i_{qs}전류)



<무부하> <부하>

그림 18. 학습을 한 후 오차 e

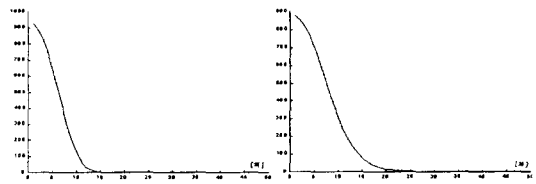
속도를 잘 추종함을 볼 수 있다. 그림 16은 에뮬레이터가 학습된 후 에뮬레이터의 오차항의 합을 역전파시켜 제어기를 최종 학습한 결과로서의 실제 유도 전동기의 속도이다. 이 때에 선형 PI 제어기와 비교하였고 부하의 유무 둘 다 본 논문에서 적용한 뉴럴 네트워크의 제어기의 오버슈트가 적음을 알 수 있어 그 성능이 우수함을 입증하였다.

이 때 학습은 50회를 실시하였다. 기준 속도에 도달시 약간의 오버슈트가 존재하지만 기준 속도를 마찬가지로 잘 추종함을 볼 수 있다. 실제로 시뮬레이션 결과 실제 속도는 1007 rpm으로서 7 rpm의 오차를 보이고 있다. 특히 부하를 가하여도 선형 제어기보다 오버 슈트가 적음을 볼 수 있으며 본 논문에서 적용한 뉴럴 네트워크 시스템이 부하에 강건함을 볼 수 있다.

그림 17은 제어기가 학습되었을 때 뉴럴 네트워크에서 출력되는 값이고 벡터 제어 유도 전동기에 가해지는 i_{qs} 전류값이다. 뉴럴 네트워크 제어기는 선형 제어기에 비해 오차가 증가함에 따라 출력 전류가 더 적음을 보여준다.

그림 18에서 기준 속도와 실제 속도와의 오차 e는 뉴럴 네트워크 제어기가 그 성능이 우수함을 볼 수 있었다.

그림 19는 50회 학습시 오차 합의 변화량을 보여주고 있다. 무부하시에는 약 15회 정도에 에러의 최소



<무부하> <부하>

그림 19. 학습시 전체 오차의 변화

값을 갖으며 부하시에는 25회 정도에서 최소치에 도달한다.

부하로 인하여 무부하시보다 약 10회 정도 더 학습을 하고 있음을 볼 수 있다. 에뮬레이터는 제어기의 학습과 비교해 볼 때 제어기 학습량의 약 1/2 정도이다. 적은 학습 회수에도 전체 뉴럴 네트워크 시스템이 빨리 수렴하는 것을 볼 수 있으며 일반적으로 뉴럴 네트워크의 학습시간이 오래 걸리는 단점이 개선됨을 볼 수 있다.

3. 결 론

시스템의 복잡한 모델링이나 수식적 해석을 줄이고 플랜트의 입,출력과 동일한 구조가 되게 뉴럴 네트워크 에뮬레이터를 설계하고 이를 이용하여 벡터 유도 전동기를 제어하기 위한 뉴럴 네트워크 제어기를 설계하였다.

본 논문에서는 플랜트의 입,출력 변수를 모의하여 학습시 성능이 좋은 에뮬레이터를 구성하였고 이 에뮬레이터를 통해 실제와 기준 속도간의 오차를 역전파시켜서 제어기의 가중치들을 학습시키도록 구성하였다. 유도 전동기의 속도 제어를 위한 뉴럴 네트워크 시스템의 학습은 오차 역전파 학습 알고리즘을 적용하고 최급 강하법을 이용하였다. 본 뉴럴 네트워크를 이용한 제어방식의 성능 평가를 위해 선형 PI 제어기를 설계하여 비교 평가하였다.

시뮬레이션 결과, 뉴럴 및 PI 제어기 둘 다 정상상태에 도달시, 약간의 오버슈트가 존재하지만 부하를 가하여 시스템의 특성이 변하는 상황에서는 본 뉴럴 네트워크를 이용한 벡터 제어 시스템의 학습 능력과 적응성으로 속도 오차의 크기가 작을 뿐만 아니라 기준 속도를 잘 추종하여 PI 제어기보다 우수함을 보여주고 있다. 일반적으로 뉴럴 네트워크 시스템은 학습이 오래 걸리는 단점이 있으나 본 논문에서는 에뮬레이터를 이용함으로서 학습 회수가 적으면서 수렴함을 볼 수 있다. 향후에는 뉴럴 네트워크 시스템을 실제 벡터 제어 유도 전동기 시스템에 적용하여 최적으로 연결 가중치를 조정하는 방법을 고찰하도록 한다.

참고문헌

[1] Kumpati S. Narendra & Kannan Parthasarthy, "Identification and Control of Dynamical Systems Using Neural Networks" *IEEE Trans. on Neural Network.*, vol. 1, No. 1 pp. 4-27, March, 1991.

[2] 김세찬, 원충연, "신경회로망을 이용한 유도전동기의 속도제어", KIEE, Vol. 45, No. 1, pp. 42-53, 1996.

[3] T. Aoyama, S. Omatu, "Design of a Self-Tuning PID Control System by Neural Networks", 電學論C, Vol. 116, No. 11, pp. 1197-1201, 平成 8年, 1998.

[4] S. Omatu, Y. Kishida, M. Yoshioka, "Neuro-Control for Single-Input Multi-Output systems", International conference on knowledge-based Intelligent Electonic Systems, Vol. 1, pp. 202-205, 1998.

[5] K. J. Hunt, D. Sbarbaro, "Neural Networks for Control System-A Survey", Automatica, Vol. 28, No. 6, pp. 1083-1112, 1992.

[6] A. Cichocki, R. Unbehauen, "Neural Networks for Optimization and Signal Processing, John Wiley, 1993.

[7] L. Tsoukalas, R. Uhrig, "Fuzzy and Neural Approaches in Engineering", John Wiley & Sons, 1997.

[8] J. S. R. Jang, C. -T. Sun, E. Mizutani "Neuro-Fuzzy and Soft Computing" Prentice Hall, 1997.

[9] M. T. Hagan, H. B. Demuth, M. Beale. "Neural Net Design", PWS publishing Company, 1996.

[10] K. Shimane, S, Tanaka, S. tanaka, "Vector controlled Induction Motor Using Neural Networks", TIEE Japan, vol. 113-D, No. 10, 1993.

[11] D. W. Novotny, T. A. Lipo, "Vector Control and Dynamics of AC Drives", Clarendon Press Oxford, 1996.

[12] C. M. Ong, "Dynamic simulation of Electric Machinery", Prentice Hall PTR, 1998.

[13] B. K Bose, "Power Electronics and AC Drives", Prentice Hall, 1986.

이 동 빈 (Dong-Bin Lee)

1992년 : 광운대 전자통신공학과 졸업
1992~1997 : (주)쌍용정공 기술 연구소 자동화팀 근무
현재 : 동 대학원 제어계측공학과 석사 과정

유 창 완 (Chang-Wan Ryu)

제9권 제4호 참조

홍 대 승 (Dae-Seung Hong)

제9권 제4호 참조

임 화 영 (Wha-Yeong Yim)

제9권 제4호 참조