

정확한 근사화 능력을 갖는 CMAC 신경망 기반 퍼지 제어기의 설계

A Design of the CMAC-based Fuzzy Logic Controller with an Accurate Approximation Ability

김 대 진, 이한별
Daijin Kim, HanPyul Lee
동아대학교 컴퓨터공학과

요 약

본 논문은 빠른 학습과 정확한 근사 능력을 갖는 새로운 CMAC 신경망 기반 퍼지 제어기를 제안한다. 제안한 CMAC 신경망 기반 퍼지 제어기(CBFLC)는 한 학습 주기 동안 전향 및 역전파 연산시 신경망내 유닛중 극히 일부분만이 활성화되어 학습에 참가하므로 학습 시간이 매우 빠르고, 비퍼지화 연산시 소속 함수의 중심값뿐 아니라 폭을 동시에 고려하여 정확한 근사화를 얻는다. 제안한 퍼지 제어기내 입·출력 소속 함수의 중심값 및 폭 등의 구조적 파라미터들은 역전파 알고리즘에 의해 갱신된다. 제안한 CMAC 신경망 기반 퍼지 제어기를 트럭 후진 주차 문제에 적용하여 근사화 능력 및 제어 성능면에서 여러 다른 퍼지 제어기들과 비교한다.

Abstract

This paper proposes a CMAC-based FLC(Fuzzy Logic Controller) with a fast learning capability and an accurate approximation ability. The proposed CMAC-based FLC shows the fast learning speed because a small number of units in the network are involved in the forward and backward computation and provides an accurate input-output approximation because it uses both centers and widths of membership functions to compute an output. The structural parameters such as the centers and widths of the membership functions are updated in the backpropagation type. Application to the truck backer-upper control problem of the proposed CMAC-based FLC is presented. The approximation accuracy and control performance are compared with those of various FLCs.

1. 서 론

퍼지 논리 제어기는 가전 및 산업 분야의 공정 제어에 폭넓게 응용되고 있다. 특히 시스템의 특성이 복잡하여 기존의 정량적인 방법으로는 해석할 수 없거나, 얻어지는 정보가 정성적이며, 부정확하고 불확실한 경우에 있어서 기존의 제어기보다 우수한 제어결과를 나타낸다[1].

이러한 퍼지 제어기에 신경망이 갖는 학습 능력을 융합하여 보다 적용성이 강하고 제어 성능을 향상시킨 신경망 형태의 퍼지 제어기가 많이 연구되고 있다[2-4].

본 연구는 기존의 CMAC 신경망과 퍼지 제어 시스템을 결합하여 CMAC 신경망 기반 퍼지 제어기를 제안하고 퍼지 제어기가 갖는 입·출력 근사화의 정확도를 향상시키고자 비퍼지화 연산시 소속 함수의 중심값 뿐 아니라 폭을 동시에 고려하여 정확한 근사화를 얻고자 한다. 나아가, 제안한 CMAC 신경망 기반 퍼지 제어기내 입·출력 소속 함수의 구조적 파라미터(중심, 폭)를 갱신하는 새로운 역전파 알고리즘을 제시한다.

본 논문의 구성은 다음과 같다. II장에서는 CMAC 신경망의 구조와 동작 원리를 설명한다.

III장에서는 제안한 CMAC 신경망 기반 퍼지 제어기의 구조와 동작 원리를 설명한다. IV장에서는 제안한 CMAC 신경망 기반 퍼지 제어기를 구조적 파라미터를 결정하는 학습 알고리즘에 대해 설명한다. V장에서는 제안한 CMAC 신경망 기반 퍼지 제어기를 트럭 후진 주차 문제에 적용하여 주행 성능 면에서 다른 퍼지 제어기와 비교한다. 마지막으로 결론과 앞으로의 연구 방향을 언급한다.

2. CMAC 신경망의 구조 및 동작 원리

CMAC 신경망은 인간의 소뇌를 단수 모델링한 신경망으로 그림 1은 Albus가 제안한 모델을 나타낸 것이다[5]. 입력 공간내 한 입력은 실세계의 N 개의 센서로부터 받아들인 N 차원 벡터로 표현된다. 이들 입력은 개념적 메모리내 C 개의 점들로 매핑되는데 입력 공간 내에서 서로 유사한 값을 가질수록 개념 메모리상에 C 개의 점들간에 중첩이 많이 일어나도록 매핑된다. 만약 두 입력이 입력 공간상에 멀리 떨어져 있다면 개념 메모리상에 C 개의 점들간에 중첩이 일어나지 않게 (no generalization) 된다. 요구되는 가상 메모리는 입

력 공간내 모든 가능한 입력의 조합만큼의 크기를 요구하므로 매우 방대하다. 그러나, 일반적으로 학습 과정이 입력 공간내 모든 가능한 입력에 대해

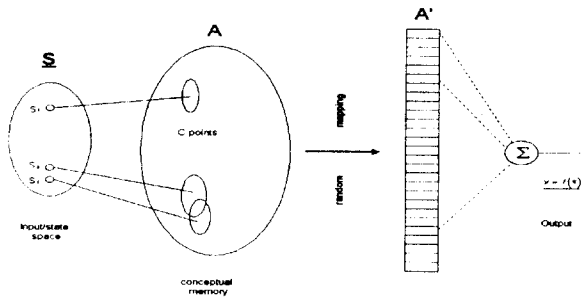


그림 1 CMAC 신경망의 구조

일어나지는 않으므로 필요로 하는 실제 물리적 메모리의 크기는 훨씬 적다. Albus는 이들 가상적 메모리와 물리적 메모리간의 매핑을 매핑후 균일한 주소 분포를 갖도록 의사 랜덤 해싱 함수에 의해 수행하였다. CMAC 신경망 출력은 입력에 의해 활성화된 물리적 메모리에 저장된 가중치의 평균에 의해 결정된다. 따라서 다른 신경망의 경우 비

하여 퍼지화를 수행하는 것으로, i 번째 입력 x_i ($i=1,2,\dots,n$)에 대해 서로 다른 반응 특성을 여러 개의 퍼지 센서 s_{ij} ($j=1,2,\dots,|x_i|$, 여기서 $|x_i|$ 는 x_i 가 갖는 퍼지 센서의 개수를 나타낸다.)로 구성된다. 각 유닛(퍼지 센서)이 갖는 반응 특성은 원래의 CMAC 신경망의 경우 그림 3의 왼쪽에 보인 것과 같이 $[0,1]$ 의 on-off 특성을 나타내지만 제안한 CBFLC는 그림 3의 오른쪽에 나타난 것과 같은 $(0,1)$ 사이의 비선형 종모양의 반응 특성을 갖는다.

따라서 각 퍼지 센서 s_{ij} 는 i 번째 입력 x_i 에 대한 j 번째 소속 함수가 갖는 소속 값 $\mu_{s_{ij}}(x_i)$ 를 나타내고 이들 소속 함수는 중심과 폭 (c_{ij}, w_{ij})에 의해 결정되어진다. 인접하는 유닛끼리 중첩도는 최대 2인 경우 한 입력에 대해 한 변수당 최대 2개의 퍼지화된 값이 얻어진다.

CMAC 신경망이 갖는 블록화 현상을 줄여 근사화 능력을 향상시키고자 그림 4와 같은 다층 구조를 갖는 퍼지 센서층을 사용할 수 있다. 만약 각 입력이 c 개의 층으로 구성되고 각 입력내 유닛간 중첩도를 d 이라고 하면, 한 n 차원 입력 (x_1, x_2, \dots, x_n) 가 입력 센서층에 들어오면, 최대 $d^n \cdot c$ 개의 퍼지 센서가 반응하게 된다. 그림 5는 중첩도가 1인 경우(즉 각 유닛이 서로 중첩되지 않을 때) 4개층으로 구성된 입력 센서에 한 입력 (x_1, x_2) 가 들어올 때, 서로 다른 위치에 놓인 4개 층에서 각기 다른 반응 출력을 나타내고 있음을 보여준다[5].

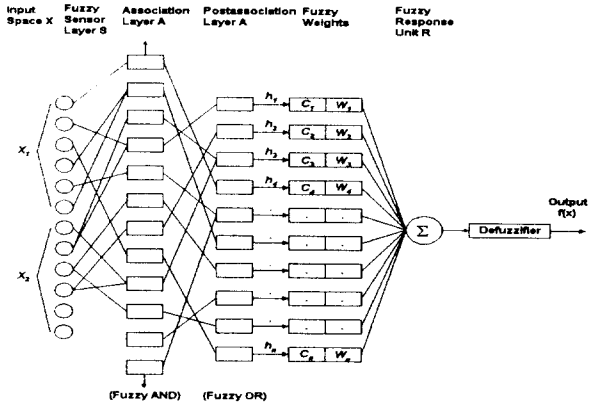


그림 2 제안한 CMAC 신경망 기반 퍼지 제어기 구조

선형성이 각 뉴런이 갖는 비선형성(sigmoid 또는 threshold 함수)에 얻어지는 것과는 달리 CMAC 신경망은 입·출력간의 비선형성을 입·출력간의 매핑에 의해 얻어진다.

3. CMAC 신경망 기반 퍼지 제어기

입력 공간내 한 입력이 N 차원 벡터 $x = (x_1, x_2, \dots, x_n)$ 으로 나타내어지고, 이에 대응하는 출력을 $y = f(x)$ 라고 할 때 제안한 CBFLC는 다음 세 개의 연속적인 매핑에 의해 입·출력사이의 비선형 함수 관계를 근사화 시킨다[6].

$$\begin{aligned} \Theta: X &\rightarrow S, \\ \Phi: S &\rightarrow A, \\ \Psi: A &\rightarrow P, \end{aligned} \quad (1)$$

여기서, $S, A,$ 및 P 는 각각 여러 유닛으로 구성된 3개의 층을 나타낸다. 그림 2은 제안한 CMAC 신경망 기반 퍼지 제어기의 구조를 나타낸 것이다.

위 그림에서 첫 번째 매핑 ($\Theta: X \rightarrow S$)은 한 crisp 입력을 받아들여 S 층내의 퍼지 센서를 이용

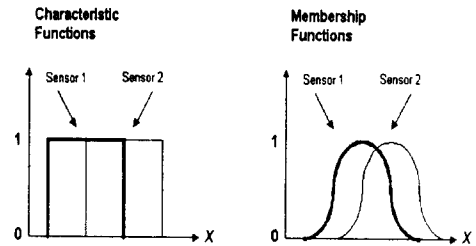


그림 3 입력 센서의 반응 특성

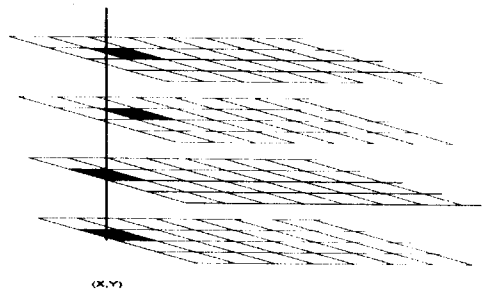


그림 4 다층 구조를 갖는 입력 센서 구조

두 번째 매핑 ($\Phi: S \rightarrow A$)은 각 입력 변수당 하나씩 전체 n 개의 퍼지 센서 출력을 받아들여 이들 중 최소값을 선택하여 출력으로 내보내는 퍼지 AND 연산을 수행한다.

세 번째 매핑 ($\Psi: A \rightarrow P$)은 활성화된 퍼지 AND 유닛 출력이 다음의 P 층내 각 유닛에 임의

로 연결되며, P층내 각 유닛에서는 들어오는 입력 값중 최대를 선택하는 퍼지 OR 연산을 수행한다.

P층내 한 퍼지 OR 유닛 출력 $\mu_{p_i}(x)$ 은 중심과 폭 (y_i, s_i) 를 갖는 종모양의 소속함수 $\mu_w(y)$ 를 갖는 가중치를 저장하는 물리적 메모리와 퍼지 규칙 관계 $\mu_{p_i}(x) \rightarrow \mu_w(y) (i=1, 2, \dots, |P|)$ 를 형성한다. 한 주어진 입력에 대해 활성화된 퍼지 OR 유닛을 P^* 라고 하면 퍼지 응답 유닛 R의 출력 $\mu_R(y)$ 는 아래 식과 같이 나타낼 수 있다.

$$\mu_R(y) = \bigvee_{p_i \in P^*} (\mu_{p_i}(x) \wedge \mu_w(y)) \quad (2)$$

여기서 \wedge 는 퍼지 AND로 흔히 min 연산이 사용되며, 위 연산의 $\mu_{p_i}(x) \wedge \mu_w(y)$ 는 퍼지 규칙의 전단부와 후단부의 상호 연관 정도를 나타내고, $\bigvee_{p_i \in P^*}$ 는 퍼지 OR로 흔히 max 연산이 사용되므로 이 경우 얻어진 최소값중 가장 최대값을 선택하는 것을 의미한다.

얻어진 퍼지 응답 $\mu_R(y)$ 는 출력 공간내 여러 곳에 분포하게 되므로 이들로부터 단일 응답 \bar{y} 를 얻기 위해 아래 식과 같은 COG(Center of Gravity) 비퍼지화 과정[7]이 요구된다.

$$\bar{y} = \frac{\int_Y y \cdot \mu_R(y) dy}{\int_Y \mu_R(y) dy} \quad (3)$$

그러나, 위 식은 적분을 포함하므로 계산량이 많아 흔히 이를 아래 식과 같이 이산적 연산에 의해 근사화 한다.

$$\bar{y} = \frac{\sum_{i=1}^{|P^*|} y_i \cdot \mu_R(y_i)}{\sum_{i=1}^{|P^*|} \mu_R(y_i)} \quad (4)$$

위 식에서 $|P^*|$ 는 활성화된 퍼지 OR 유닛의 수이고 y_i 와 $\mu_R(y_i)$ 는 각각 i 번째 활성화된 퍼지 OR 유닛에 대응하는 가중치 소속 함수의 중심값과 중심 y_i 에서 퍼지 소속 함수값이다. 그러나, 식 (3)을 식 (4)에 의해 근사화하는 경우 연산의 복잡도를 줄이기 위해 출력 소속 함수의 중심에서의 단일 출력값(Singleton)을 사용하여 비퍼지화를 행하는데, 이는 비퍼지화값에 많은 오차를 유발하여 정확한 제어를 어렵게 만들 수 있다. 이를 극복하기 위해 출력 변수가 갖는 소속 함수 중심에서의 단일 출력값만을 고려하는 대신 소속 함수가 갖는 면적을 고려하여 비퍼지화값을 연산할 것을 제안하였는데[8], 본 논문에서는 아래 식과 같이 출력 소속 함수가 갖는 면적을 소속 함수의 높이와 폭의 곱($\mu_R(y_i) \cdot s_i$)에 의해 근사화시키는 방안을 고려하였다.

$$\bar{y} = \frac{\sum_{i=1}^{|P^*|} y_i \cdot \mu_R(y_i) \cdot s_i}{\sum_{i=1}^{|P^*|} \mu_R(y_i) \cdot s_i} \quad (5)$$

4. CMAC 신경망 기반 퍼지 제어기의 학습

기존의 CMAC 신경망은 신속한 학습을 위해 각 유닛이 선형적으로 동작한다고 가정하고 각 유닛의 학습을 위해 잘 알려진 LMS 적응 규칙을 사용

한다. 따라서, CMAC 신경망의 입 · 출력 근사화 능력에는 블록화 현상과 같은 단점이 나타난다. 이를 개선하기 위해 본 연구에서는 다층 해상도 구조와 각 유닛이 비선형 특성을 갖는다고 가정하였다. 그리고, 제안한 CBFLC는 비록 각 층에서 동작하는 기능은 서로 다르지만 정확히 다층 퍼셉트론 구조를 가지며 갖는다. 따라서, 논문에서는 다층 퍼셉트론 구조를 갖는 퍼지 제어를 학습시키는 데 역전과 알고리즘을 사용한 Lin과 Lee[2]의 방법을 다음과 같은 두 가지 사항을 고려하여 수정하고자 한다. 먼저, 기존의 다층 퍼셉트론 신경망에 쓰이는 역전과 알고리즘은 출력단의 오차를 각 유닛의 가중치를 갱신하는데 사용하지만, 제안한 CBFLC는 S층내의 각 유닛이 갖는 입력 소속 함수를 정의하는 중심값과 폭과 P층내의 각 유닛이 갖는 출력 소속 함수를 정의하는 중심값과 폭을 결정하는 것이 요구되므로 이러한 입 · 출력 소속 함수의 구조적 파라미터들을 갱신하는데 사용되어지도록 한다. 다음, 기존의 MLP 신경망에 쓰이는 역전과 알고리즘은 각 층간의 유닛을 연결하는 가중치가 모두 갱신되는 것에 비해, CBFLC에서는 비슷한 입력은 비슷한 출력을 나타내고 서로 다른 입력은 다른 출력을 나타내도록 국부적으로 반응하므로 전체 유닛중 입력에 반응한 유닛만 갱신되어지도록 한다. 따라서, 전체적 일반화 특성을 갖는 역전과 학습보다 매 iteration마다 주어진 입력에 대해 활성화된 유닛만이 학습에 참여하므로 학습 시간이 훨씬 단축되는 장점이 있다. 다음은 제안한 CBFLC를 학습시키는 과정을 자세히 설명한 것이다.

한 학습 샘플 (x^t, y^t)에 대해, 제안한 CBFLC는 입력 x 에 대해 식 (5)에 의해 비퍼지화 출력 \bar{y}^t 를 나타낸다. 본 연구에서는 식 (6)과 같이 N개로 이루어진 전체 학습 샘플에 의한 출력단의 제어 오차 E를 역전과시키는 배치 학습[9]을 사용하였다.

$$E = \sum_{t=1}^N \frac{1}{2} (y^t - \bar{y}^t)^2 = \frac{1}{2} \sum_{t=1}^N \left(y^t - \frac{\sum_{i=1}^{|P^*|} y_i \cdot \mu_R(y_i) \cdot s_i}{\sum_{i=1}^{|P^*|} \mu_R(y_i) \cdot s_i} \right)^2 \quad (6)$$

여기서 y^t 는 t 번째 원하는 출력이고, \bar{y}^t 는 t 번째 입력 x^t 에 대한 CBFLC의 비퍼지화 출력값이다.

P층내 활성화된 유닛의 출력 소속 함수의 중심과 폭 (y_i, s_i) 는 generalized 델타 규칙에 의하면 다음과 같이 갱신된다.

$$y_i(t+1) = y_i(t) - \eta_i^y \frac{\partial E}{\partial y_i} \quad (7)$$

$$s_i(t+1) = s_i(t) - \eta_i^s \frac{\partial E}{\partial s_i}$$

여기서, η_i^y 와 η_i^s 는 각각 i 번째 출력 소속 함수의 중심과 폭의 학습율을 나타낸다.

위 식내 중심과 폭에 대한 오차의 변화율을 chain 규칙과 간단한 미분 규칙을 이용하여 구하면 아래 식과 같다.

$$\frac{\partial E}{\partial y_i} = \sum_{t=1}^N \frac{\partial E^t}{\partial y^t} \frac{\partial y^t}{\partial y_i} = \sum_{t=1}^N (y^t - \bar{y}^t) \frac{\mu_R(y_i) \cdot s_i}{\sum_{i=1}^{|P^*|} \mu_R(y_i) \cdot s_i} \quad (8)$$

$$\begin{aligned} \frac{\delta E}{\delta s_i} &= \sum_{j=1}^N \frac{\delta E^j}{\delta y^j} \frac{\delta \bar{y}^j}{\delta s_i} \\ &= \sum_{j=1}^N (y^j - \bar{y}^j) \frac{\mu_R(y_i) \left(\sum_{j=1}^N \mu_R(y_j) \cdot s_j \cdot (y_i - y_j) \right)}{\left(\sum_{j=1}^N \mu_R(y_j) \cdot s_j \right)^2} \quad (9) \end{aligned}$$

따라서, 출력 소속 함수의 중심과 폭 (y_i, s_i) 을 갱신시키는 학습 규칙은 다음과 같다.

$$\begin{aligned} y_i(t+1) &= y_i(t) - \eta_i^y \cdot \sum_{j=1}^N (y^j - \bar{y}^j) \\ &\quad \frac{\mu_R(y_i) \cdot s_i}{\sum_{j=1}^N \mu_R(y_j) \cdot s_j} \\ s_i(t+1) &= s_i(t) - \eta_i^s \cdot \sum_{j=1}^N (y^j - \bar{y}^j) \quad (10) \\ &\quad \frac{\mu_R(y_i) \left(\sum_{j=1}^N \mu_R(y_j) \cdot s_j \cdot (y_i - y_j) \right)}{\left(\sum_{j=1}^N \mu_R(y_j) \cdot s_j \right)^2} \end{aligned}$$

출력단에서 얻어진 제어 오차 E는 P층내 활성화된 유닛을 통해 역전파되는데 P층내 i번째 활성화된 유닛이 갖는 오차 E_p^i 는 유닛이 갖는 소속 함수에 따라 다음 식에 의해 결정된다.

$$\begin{aligned} E_p^i &= -\frac{\delta E}{\delta \mu_R(y_i)} = -\sum_{j=1}^N \frac{\delta E}{\delta y^j} \frac{\delta \bar{y}^j}{\delta \mu_R(y_i)} \\ &= -\sum_{j=1}^N (y^j - \bar{y}^j) \\ &\quad \left\{ \frac{y_i \cdot s_i \left(\sum_{j=1}^N \mu_R(y_j) \cdot s_j \right) - \left(\sum_{j=1}^N y_j \cdot \mu_R(y_j) \cdot s_j \right) \cdot s_i}{\left(\sum_{j=1}^N \mu_R(y_j) \cdot s_j \right)^2} \right\} \quad (11) \end{aligned}$$

P층으로 역전파된 오차 E_p^i 는 P층을 활성화시킨 A층의 퍼지 OR 유닛으로 그대로 역전파된다. 따라서 A층내 활성화된 퍼지 OR 유닛이 갖는 오차 E_A^i 는 E_p^i 와 같다. A층으로 역전파된 오차 E_A^i 는 A층내 활성화된 퍼지 AND 유닛으로 들어오는 퍼지 센서 입력중 가장 작은 값을 갖는 퍼지 센서쪽으로 역전파된다. i번째 입력에 대한 j번째 퍼지 센서가 갖는 소속 함수가 종모양의 (bell-shaped)의 $f_{ij} = \exp\left(-\frac{1}{2} \frac{(x_i - x_{ij})^2}{s_{ij}^2}\right)$ 를 갖는다고 가정하면, 퍼지 센서의 구조적 파라미터는 generalized 델타 규칙에 의하면 다음과 같이 갱신된다.

$$\begin{aligned} x_{ij}(t+1) &= x_{ij}(t) - \eta_{ij}^x \frac{\partial E}{\partial x_{ij}} \\ s_{ij}(t+1) &= s_{ij}(t) - \eta_{ij}^s \frac{\partial E}{\partial s_{ij}} \quad (12) \end{aligned}$$

여기서, η_{ij}^x 와 η_{ij}^s 는 각각 i번째 입력에 대한 j번째 입력 소속 함수의 중심과 폭의 학습율을 나타낸다. 위 식내 중심과 폭에 대한 오차의 변화율을 chain 규칙과 간단한 미분 규칙을 이용하여 구하면 아래 식과 같다.

$$\begin{aligned} \frac{\partial E}{\partial x_{ij}} &= \frac{\partial E}{\partial f_{ij}(x)} \frac{\partial f_{ij}(x)}{\partial x_{ij}} \\ &= -\sum_k e_k \cdot f_{ij}(x_i) \cdot \frac{(x_i - x_{ij})}{s_{ij}^2} \quad (13) \end{aligned}$$

$$\begin{aligned} \frac{\partial E}{\partial s_{ij}} &= \frac{\partial E}{\partial f_{ij}(x)} \frac{\partial f_{ij}(x)}{\partial s_{ij}} \\ &= -\sum_k e_k \cdot f_{ij}(x_i) \cdot \frac{(x_i - x_{ij})^2}{s_{ij}^3} \quad (14) \end{aligned}$$

여기서,

$$e_k = \begin{cases} E_A^k & j\text{번째 센서 출력} \\ k\text{번째 퍼지 AND 유닛의 입력중 최소일때} \\ 0 & \text{기타} \end{cases}$$

따라서, 입력 소속 함수의 중심과 폭 (x_{ij}, s_{ij}) 을 갱신시키는 학습 규칙은 다음과 같다.

$$\begin{aligned} x_{ij}(t+1) &= x_{ij}(t) + \eta_{ij}^x \cdot \sum_k e_k \cdot f_{ij}(x_i) \cdot \frac{(x_i - x_{ij})}{s_{ij}^2} \\ s_{ij}(t+1) &= s_{ij}(t) + \eta_{ij}^s \cdot \sum_k e_k \cdot f_{ij}(x_i) \cdot \frac{(x_i - x_{ij})^2}{s_{ij}^3} \quad (15) \end{aligned}$$

5. 실험 결과 및 분석

제안한 CBFCLC가 제대로 동작하는지를 트럭 후진 주차 문제에 테스트해 보았다. 트럭의 현재 위치는 (x, y, ϕ) 에 의해 결정되고, 트럭의 후진 주행 제어는 트럭 진행 방향과 x축간의 각도 ϕ 와 핸들과 트럭의 중심축 간의 각도인 θ 에 의해 결정된다. 트럭이 움직이는 운동 방정식은 아래와 같이 나타내어진다[10].

$$\begin{aligned} x(t+1) &= x(t) + \cos[\phi(t) + \theta(t)] + \sin[\theta(t)] \cdot \sin[\phi(t)] \\ y(t+1) &= y(t) + \sin[\phi(t) + \theta(t)] - \sin[\theta(t)] \cdot \sin[\phi(t)] \\ \phi(t+1) &= \phi(t) - \sin^{-1} \left[\frac{2 \sin(\theta(t))}{b} \right] \quad (16) \end{aligned}$$

여기서 b는 트럭의 길이이며, 본 논문에서는 $b = 4$ 로 하였다. 만약 트럭과 주차대까지의 거리가 충분하다면 트럭이 $x = 10, \phi = 90^\circ$ 가까이 오면 트럭을 곧장 후진하기만 하면 되기 때문에 변수 y를 퍼지 입력 변수 (x, y, ϕ) 에서 뺄 수 있다. 그러므로 트럭 주차 제어 문제는 주어진 공간내 $(0 \leq x \leq 20, -90^\circ \leq \phi \leq 270^\circ)$ 임의의 초기 위치 (x_0, ϕ_0) 에서 가능하면 신속·정확하게 주차대 $(x = 10, \phi = 90^\circ)$ 쪽으로 후진하도록 바퀴 각도 $\theta (-40 \leq \theta \leq 40)$ 를 제어하는 것이 요구된다.

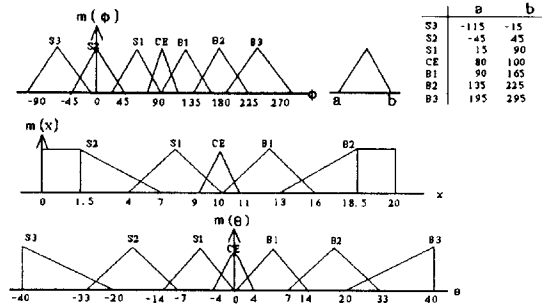


그림 5 트럭 후진 주차 제어에 사용된 소속 함수와 퍼지 규칙 베이스

그림 5는 Wang과 Mendel[11]이 사용한 퍼지 제어기의 입·출력 변수의 소속 함수와 퍼지 제어를 위한 퍼지 규칙 베이스를 나타낸 것이다.

본 연구에서 구현한 CBFCLC의 구조적 사양 및 학습 파라미터가 표 1에 나타나 있다. 여기서 입력 변수 x 와 ϕ 의 소속 함수 개수는 Wang이 사용한 퍼지 제어기[11]의 경우와 공평한 비교를 위해 각각 5개, 7개로 선택하였다. 기존의 CMAC 신경망

의 경우 가상 메모리의 크기가 ∞ 개였지만, 각 입력의 범위가 $[0, |x_i|]$ 이고 해상도가 δ 이면 각 입력을 $N_\delta = \frac{|x_i|}{\delta}$ 구간으로 나눌 수 있으므로 n 차원 입력의 경우 요구되는 가상 메모리의 크기는 $(N_\delta)^n$ 이므로 충분하므로 본 연구에서는 $256 \times 256 = 65,536$ 를 선택하였다.

제한한 CMAC 신경망 기반 퍼지 제어기와 MLP 신경망 기반 퍼지 제어기와 학습 시간을 이론적으로 비교해 보면 다음과 같다. 먼저 MLP 신경망 기반 퍼지 제어기의 경우 학습 회수를 N_{MLP} , 제한한 CMAC 신경망 기반 퍼지 제어기의 학습 회수를 N_{CMAC} 이라 하자. 앞서 언급하였듯이, MLP 신경망 기반 퍼지 제어기는 입력 공간의 전역적 일반화 특성을 추구하는데 비해 CMAC 신경망 기반 퍼지 제어기는 입력 공간의 국부적 일반화 특성을 요구하므로 학습이 훨씬 쉬워 N_{CMAC} 이 N_{MLP} 보다 훨씬 적음을 알 수 있다.

표 1 제안한 CBFLC의 구조 및 학습 파라미터

	입력 변수당 층수 = 1	입력 변수당 층수 = 4
입 · 출력 변수	$(x, \phi; \theta)$	$(x, \phi; \theta)$
입 · 출력 변수의 소속 함수 개수	(5, 7; 1024)	(20, 28; 4096)
활성화되는 입 · 출력 변수의 소속 함수 개수	(2, 2; 4)	(8, 8; 16)
가상적 메모리 크기	65,536	65,536
물리적 메모리 크기	1,024	4,096
학습 샘플수	236	236
입 · 출력 변수의 학습율 η	0.5	0.5
학습 횟수	200	200

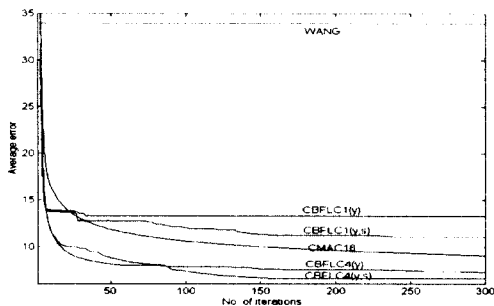


그림 6 여러 가지 제어기의 근사화 능력 비교

그림 6는 본 연구에서 제안한 CMAC 신경망 기반 퍼지 제어기를 트럭 후진 주차 문제에 적용한 경우의 학습에 따른 오차 곡선을 나타낸다. 이 그림으로부터 학습 횟수가 200회 이상 지나면 학습

오차가 수렴되는 것을 알 수 있고, 비슷한 실험에서 LIN[2]이 제안한 MLP 신경망 기반 퍼지 제어기는 약 2,000회 이상 학습하여야 학습 오차가 수렴되는 것과 비교해보면[2] 실험적으로 대략 $N_{CMAC} \approx 10^{-1} N_{MLP}$ 임을 알 수 있다. 다음으로 매 학습마다 요구되는 연산을 서로 비교해본다. 두 퍼지 제어기가 각층에서 갖는 유닛의 수를 각각 M_{MLP} 개와 M_{CMAC} 개로 서로 동일하다고 가정하자. 순방향 또는 역전파 연산시 두 퍼지 제어기의 각 유닛에서 수행되는 연산 과정은 기본적으로 동일하지만 연산에 참여하는 유닛의 수는 크게 다르다. MLP 신경망 기반 퍼지 제어기의 경우 한 입력에 대해 각 층내의 모든 유닛이 연산에 참여하지만 CMAC 신경망 기반 퍼지 제어기의 경우 중의 유닛중 주어진 입력에 대해 극히 적은 일정한 수의 유닛만이 연산에 참여하므로 매 iteration시 요구되는 연산의 양은 CMAC 신경망 기반 퍼지 제어기가 훨씬 적음을 알 수 있다. 이상의 두 가지 사실로부터 제안한 CMAC 신경망 기반 퍼지 제어기가 MLP 신경망 기반 퍼지 제어기보다 훨씬 신속하게 학습함을 알 수 있다.

다음 제안한 CMAC 신경망 기반 퍼지 제어기의 입 · 출력간 근사화 능력을 여러 가지 다른 근사화 방법과 비교해본다. 이를 위해 WANG[11]이 사용한 236개의 학습 샘플을 이용하여 출력에서의 평균 오차 $E = \sum_{i=1}^{236} \frac{1}{2} (y' - \bar{y})^2$ 를 사용하였다. 그림 6에서 수직축은 평균 오차를 나타내며, 수평축은 여러 가지 다른 근사화 방법들을 나타내는 것으로 각각 학습 과정이 없이 설계자의 직관에 의해 얻어진 WANG의 퍼지 제어기, CMAC_16 신경망 (입력 layer 수 = 16), 기존의 비퍼지화기를 사용한 CBFLC(y) (출력 소속 함수의 폭이 일정, $s_i = s = \text{constant}$), 제안한 비퍼지화기를 갖는 CBFLC(y,s) (출력 소속 함수의 중심과 폭이 가변적)를 의미하며, 마지막 두 경우에 대해서는 각각 입력 센서층의 개수가 $c=1$ 과 $c=4$ 인 경우 두 가지에 대해 실험을 수행하였다. 실험 결과, 학습이 진행된 시점 (학습 회수 = 300)에서 평균 오차는 CBFLC_4(y,s), CBFLC_4(y), CMAC_16, CBFLC_1(y,s), CBFLC_1(y), WANG's FLC의 순서로 각각 6.60, 7.22, 9.07, 11.08, 13.31, 34.0임을 알 수 있었다. 따라서, 비학습보다는 학습한 경우, 같은 CBFLC인 경우 입력 변수가 갖는 layer가 많을수록, 비퍼지화 연산시 출력 소속 함수의 중심값만 학습시키는 것보다는 중심값과 폭을 동시에 학습시키는 것이 보다 우수한 근사화 능력을 나타냄을 알 수 있었다.

다음, 제안한 CMAC 신경망 기반 퍼지 제어기의 제어 성능을 여러 가지 다른 제어기와 비교해본다. 이를 위해 본 연구에서는 모델 트럭이 100개의 임의의 랜덤한 초기 위치 (x_i, ϕ_i) 에서 목표점 $(x = 10, \phi = 90^\circ)$ 까지 도달하는데 걸린 평균 주행 거리를 구해서 비교하였다. 평균 거리는 $d = \frac{1}{N_r} \sum_{i=1}^{N_r} \sum_{j=1}^{S_i} d_{ij}$ 에 의해 계산되는데, N_r 은 도달한 샘플 개수, S_i 는 i 번째 샘플의 도달할 때까지의 스텝수, d_{ij} 는 i 번째 샘플의 j 번째 스텝에서의 움직임 거리로 $d_{ij} = \sqrt{(x_{i,j+1} - x_{i,j})^2 + (y_{i,j+1} - y_{i,j})^2}$ 를 나타낸다.

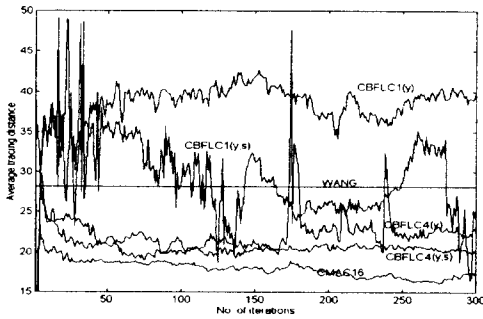


그림 7 여러 가지 제어기의 제어 성능 비교

그림 7에서 수직축은 평균 주행 거리를 나타내며, 수평축은 여러 가지 제어기들을 나타내는 것으로 각각 학습 과정이 없이 설계자의 직관에 의해 얻어진 WANG의 퍼지 제어기, CMAC_16 신경망 (입력 layer 수 = 16), 기존의 비퍼지화기를 사용한 CBFLC(y) (출력 소속 함수의 폭이 일정, $s_i = s = \text{constant}$), 제안한 비퍼지화기를 갖는 CBFLC(y,s) (출력 소속 함수의 중심과 폭이 가변적)를 의미하며, 마지막 두 경우에 대해서는 각각 입력 센서층의 개수가 $c=1$ 과 $c=4$ 인 경우 두 가지에 대해 실험을 수행하였다. 실험 결과, 학습이 진행된 시점 (학습 회수 = 300)에서 평균 주행 거리는 CMAC_16, CBFLC_4(y,s), CBFLC_4(y), CBFLC_1(y,s), WANG's FLC, CBFLC_1(y)의 순서로 각각 17.73, 20.72, 21.91, 22.53, 24.09, 27.89, 39.92임을 알 수 있었다. 이 결과로부터, (1) 16층을 갖는 CMAC이 가장 짧은 평균 주행 거리를 나타내는데, 이로부터 입력 퍼지 센서의 층수가 제어 성능을 매우 큰 영향을 미치는 것을 알 수 있었고, (2) 같은 CBFLC이라 하더라도 비퍼지화 연산시 출력 소속 함수의 중심만 고려하는 것보다 소속 함수가 갖는 면적을 고려하는 것이 더욱 우수한 제어 성능을 보였으며, (3) 학습을 하지 않고 전문가의 주관적 경험에 의해 소속 함수와 퍼지 제어 규칙을 결정된 WANG의 퍼지 제어기의 제어 성능은 비교적 나쁜 것으로 나타났다.

6. 결론

본 논문은 기존의 MLP 신경망 기반 퍼지 제어기가 갖는 학습 시간이 오래 걸리는 단점과 기존의 CMAC 신경망이 갖는 입·출력 근사화 정도의 부정확성을 개선하고자 이들을 결합한 CMAC 신경망 기반 퍼지 제어기를 제안한다. MLP 신경망 기반 퍼지 제어기는 입력 공간의 전역적 일반화 특성을 추구하는데 비해 CMAC 신경망 기반 퍼지 제어기는 입력 공간의 국부적 일반화 특성을 요구하므로 학습이 훨씬 쉬워 빨리 수렴하며, 매 학습마다 요구되는 연산도 CMAC 신경망 기반 퍼지 제어기의 경우 전체 유닛중 주어진 입력에 대해 극히 적은 일정한 수의 유닛만이 연산에 참여하므로 연산량이 훨씬 적어 전체적으로 CMAC 신경망 기반 퍼지 제어기의 학습 시간이 훨씬 적음을 알 수 있다. 그리고, 기존의 CMAC 신경망이 출력값을 얻는데 연산의 복잡도를 줄이기 위해 출력 소속 함수의 중심에서의 단일 출력값 (Singleton)을 사용하여 비퍼지화를 행하는데, 이는 비퍼지화값에 많은 오차를 유발하여 정확한 제어를 어렵게 만들 수 있다. 이를 극복하기 위해 출력 변수가 갖는 소속 함수 중심에서의 단일 출력값만

을 고려하는 대신 소속 함수가 갖는 면적을 고려하여 비퍼지화값을 연산하는데, 면적을 적분기에 의해 직접 계산하면 연산 시간이 많이 걸리므로 적분기를 사용할 때에 비해 제어 결과가 크게 뒤떨어지지 않으면서 연산의 복잡도를 크게 증가시키지 않는 방안으로 출력 소속 함수가 갖는 면적을 소속 함수의 높이와 폭의 곱($\mu_R(y_i) \cdot s_i$)에 의해 근사화시키는 방안을 사용하였다.

제한한 CBFLC가 제대로 동작하는지를 트럭 후진 주차 문제에 대해 테스트해 보았다. 제한한 CMAC 신경망 기반 퍼지 제어기의 입·출력간 근사화 능력과 제어 성능을 각각 평균 오차와 평균 주행 거리면에서 WANG의 퍼지 제어기, CMAC 신경망, 기존의 비퍼지화기를 사용한 CBFLC (출력 소속 함수의 폭이 일정, $s_i = s = \text{constant}$), 제안한 비퍼지화기를 갖는 CBFLC (출력 소속 함수의 중심과 폭이 가변적) 등과 비교하였다. 실험 결과, 제안한 CBFLC가 다른 제어기보다 입·출력간 근사화 능력과 제어 성능면에서 모두 우수한 결과를 나타냄을 확인하였다.

참고 문헌

- [1] E. H. Mandani, "Application of fuzzy algorithms for control of simple dynamic plant," *IEEE Proc. Control & Science*, vol. 121, no. 12, pp. 1585-1588, Dec. 1974.
- [2] C. Lin and C. S. Lee, "Neural-Network-Based Fuzzy Logic Control and Decision System," *IEEE Trans. on Computers*, vol. 40, no. 12, pp. 1320-1336, 1991.
- [3] J.-S. Jang, "ANFIS: Adaptive-Network-Based Fuzzy Inference System," *IEEE Transactions on System, Man, and Cybernetics*, vol. 23, no. 3, pp. 665-685, 1993.
- [4] J. S. Albus, "A new approach to manipulator control: The Cerebellar Model Articulation Controller (CMAC)," *Journal of Dynamic Systems, Measurement, and Control*, pp. 220-227, Sept., 1975.
- [5] W. T. Miller, F. H. Glanz, and L. G. Kraft, "CMAC: An associative neural network alternative to backpropagation," *Proceedings of IEEE*, vol. 78, no. 10, pp. 1561-1567, Oct. 1990.
- [6] C.-C. Jou, "A fuzzy cerebellar model articulation controller," *IEEE International Conference on Fuzzy Systems*, vol. 2, pp. 1179-1186, San Diego, March, 1992.
- [7] C. C. Lee, "Fuzzy Logic in Control Systems: Fuzzy Logic Controller," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 20, no. 2, pp. 404-435, Feb. 1990.
- [8] Daijin Kim, "Improving the Fuzzy System Performance by Fuzzy System Ensemble," *Fuzzy Sets and Systems*, In Print, 1997.
- [9] H.-S. Rhee and K.-W. Oh, "A Design and Analysis of Objective Function-based Unsupervised Neural Networks for Fuzzy Clustering," *Neural Processing Letters*, vol. 4, no. 2, pp. 83-95, 1996.
- [10] Li-Xin Wang And Jerry M. Mendel,

"Generating Fuzzy Rules from Numerical Data with Applications," *USC-SIPI Report*, no.169, 1991.

- [11] Li-Xin Wang and Jerry M. Mendel, "Generating Fuzzy Rules by Learning from Examples," *IEEE Transactions on System, Man, and Cybernetics*, vol. 22, no. 6, pp. 1414-1427, Nov. 1992.