Proceedings of KFIS 2001 Spring Conference, 2001. 5. 19

컬러재현을 위한 CMAC의 뉴로퍼지 설계

CMAC Neuro-Fuzzy Design for Color Calibration

이철희 · 변오성 · 문성룡 · 임기영

원광대학교 전자공학과

Abstract

CMAC model was proposed by Albus [6] to formulate the processing characteristics of the human cerebellum. Instead of the global weight updating scheme used in the back propagation, CMAC use the local weight updating scheme. Therefore, CMAC have the advantage of fast learning and high convergence rate. In this paper, simulate Color Calibration by CMAC in color images and design hardware by VHDL-base high-level synthesis.

Key Words : Neuro-fuzzy, CMAC, Color calibration, Hardware design

1. 서 론

시스템의 복잡도가 증가할수록 모델 방정식을 표현함에 있어서 어려움을 수반할 뿐만 아니라 모델 방정식에서 발생하는 많은 계산량과 변수를 다루는 새로운 방법이 필요하게 된다.

이를 위한 비선형 모델을 구현하기 위한 예를 들면 Rosenblatt의 perceptron과 Widrow 의 Adaline, Hopfield 의 rec-urrent model등을 들 수 있다.[1][2] 많은 계산량과 시스템의 복잡도를 가지고 있는 컬러이미지의 색채 변환(Color reproduction)은 다른 매체 프린터, 스캐너, 모니터 등의 서로 다른 색 입력을 장치에 맞도록 조정하는 것으로 색채 변환 모델로는 BP(Back Propagation)이나 Rectangular type CMAC(Cerebellar Model Artic -ulation Computer)등이 있다.[3] 이중에서 CMAC는 J.S.Albus가 1970년대에 이용했으나 로봇 제어 관련 응용분야에서는 많은 제약을 받았으나 최근에 CMAC는 여러 가지 제어 응답에 의존하는 상태공간을 학습하는 데에 사용되어질 수 있음이 입증되었다. CMAC는 넓은 범위에 함수의 입력으로부터 비선형 관계를 학습할 수 있고, 학습 알고리즘은 일반적으로 적은 반복으로도 수렴하는 특성을 가지고 있다. 본 연구에서는 비선형이면서 보다 높은 자유도를 가지고 함수를 근사화 할 수 있는 B-스플라인(B-spline)함수를 리셉티브 필드(receptive field)로 적용한 CMAC를 설계하고, 특히 CMAC의 주소 생성에 PE를 사용할 수 있도록 PE를 스토릭 어레이(SA:Systolic array)의 구조를 갖는 하드웨어로 구현하고자 한다.

2. 표준 CMAC 이론

먼저 Albus에 의해서 제안된 CMAC 신경망의 구조를 살펴보면 그림 1과 같다. 그림 1-1에는 두가지의 매핑이 존재한다. 하나는 S 매핑이고 다른 하나는 P 매핑이다. 어떤 입력벡터가 리셉티브 필드로 입력되면 CMAC 신경망의 응답은 입력에 의해 활동하게된 리셉티브 필드의 응답(S 매핑)의 평균으로 구해진다(P 매핑).

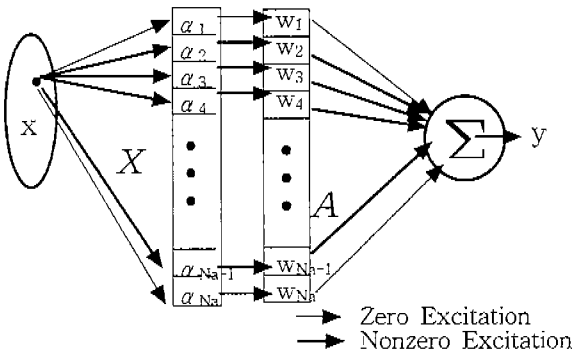

그림 1 표준 CMAC의연산과정

Fig. 1 Schematic of Standard CMAC model

3. CMAC의 구조

CMAC는 연속된 변환에 의해서 정의된다. 변환은 다음과 같다. [3]

$$S \rightarrow A \rightarrow P \qquad (1)$$

S는 입력벡터 ($S=\langle s_1, s_2, \ldots, s_N \rangle$), A는 리셉티브 필드의 벡터 어드레스, P는 출력벡터이다.

S로 표현된 입력 벡터는 CMAC연산의 첫번째 단계에서 입력 벡터 S'로 정형화(normalized)된다.

$$S' = \langle s'_1, s'_2, \ldots, s'_N \rangle = \langle int\left(\frac{s_1}{\Delta_1}\right), int\left(\frac{s_1}{\Delta_2}\right), \ldots, int\left(\frac{s_N}{\Delta_N}\right) \rangle \qquad (2)$$

각 리셉티브 필드의 폭은 $C*\Delta_j$와 같다. 이 과정에서 원래의 입력 공간은 정형화된 입력공간(normalized input space)으로 바뀐다.

다음 단계에서 이 S'를 이용하여 리셉티브 필드의 벡

터 어드레스A_i를 얻는다. [4]

$$A_i=\langle s'_1-[(s'_1-i)\%C],\ s'_2-[(s'_2-i)\%C],\ \ldots,\ s'_N-[(s'_N-i)\%C]\rangle$$
$$i=1,2,\ldots,C$$
$$=\langle a_{i1},\ a_{i2},\ \ldots,\ a_{iN}\rangle \quad (3)$$

%는 Modulus 연산자, I 는 C 개의 리셉티브 필드의 병렬 층의 순서, A_j 는 각 층에서 발현된 정형화된 N-차원 어드레스이다.

N차 공간에서 리셉티브 필드의 수는 매우 크므로 가상공간으로 변환할 필요가 있다. 다음으로 CMAC는 물리번지(A'_i)로 부터 얻어서 조정가능한 가중치(Adjustable Weight)를 가져오기 위하여 식 (4)를 수행한다.

$$A'_i=h(a_{i1},\ a_{i2},\ \ldots,\ a_{iN}) \quad (4)$$

여기서 h(...)는 크기가 M인 물리메모리(Physical Memory)의번지를 발생시켜주는 pseudorandom 해싱 함수(Hashing function)이고, 마지막으로 CMAC는 다음의 식으로 출력을 얻는다. [4]

$$y(S)=\frac{1}{C}\sum_{i=1}^{C} W[A'_i] \quad (5)$$

학습은 가중치(weight) ω를 수정함으로써 이루어질 수 있으며, 학습 과정은 식 (6)과 같다. [5][6]

$$w_j = w_j+\beta(y_d-y) \quad (6)$$

여기서, y_d는 요구되는 출력값, y는 CMAC의 출력, β는 양의 학습율이다.

β가 1이면 y_d와 정확히 일치하는 출력을 만들고, 0.5이면 y_d에 정확히 1/2씩 접근한다.

연산과정에서 입력벡터는 A로 매핑되며, 매핑과정 에서 A'는 A중에서 0이 아니면서, 활성화된 집합이다. 출력 P는 이 A의 가중치를 덧셈하여 얻는다.

4. 주소 추출 과정과 학습

위 에서 2 차원의 입력 공간에서 CMAC를 고려하였다. 저 차원에서는 고려되지 않았던 것이 차원이 높아질수록 발생하는데 그 중 하나가 이웃이다.

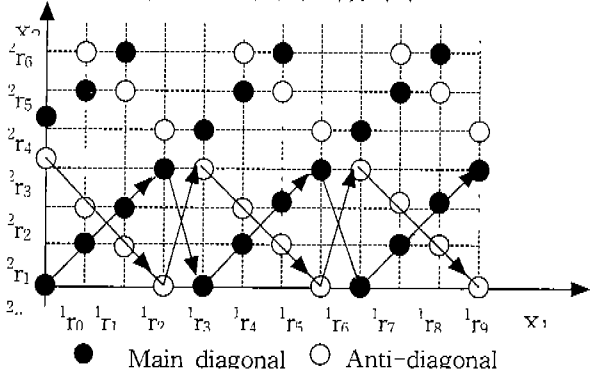

그림 2. 주,부 대각성분을 고려한 가중치 주소생성
Fig. 2 Main and Sub diagonal weight addressing

이것은 차원에 따라서 달라지게 되는데, 어소시에이션 셀이 4개인 경우 16개의 셀이 2 차원에 존재하게 되고, 3차원인경우에는 64개의 셀이 존재하게 된다. 이 주소공간을 해시 코딩에 어떻게 관련시키는가의 문제가 대두되는데, 이것에 대한 것이 이웃이다.

이웃관계에 따라서 해시 코딩이 결정되고, 이에 따른 해싱함수와 보다 적은 메모리로의 매핑을 위한 해싱이 수행되게 된다. 그림 2는 어드레스 생성을 위한 이웃의 상태를 그린 것이다.[8]

우선 B-스플라인 계수를 구하고, 다음으로 가중치(S →A 매핑)를 구해야 한다. 빠른 어드레스 발생기는 시스템의 성능을 향상시킬 수 있다. 어소시에이션 셀로부터 얻은 주소는 매우 방대함으로 해시 코드를 사용하여 메모리를 줄일 필요가 있다. 그 이유는 전제 cell 공간이 매번 사용되는 것이 아니고 4개의 cell만이 입력공간에서 활성화(active)되기 때문이다. 여기에서 해싱 함수가 사용되는데 사용한 해싱 함수는 다음과 같다.

해싱함수는 직접 가중치 셀 주소 매핑 공식을 확장하여 사용한다. 이는 식 (4-1)과 같다. [10]

$$PN_j=\sum_{m=1}^{M-1}\left(\prod_{i=m+1}^{M}\left(1+\left\lceil\frac{Q_i-J_{i,j}}{G}\right\rceil\right)\left\lceil\frac{q_m-J_{m,j}}{G}\right\rceil\right)+\left\lceil\frac{q_M-J_{M,j}}{G}\right\rceil \quad (7)$$

이 확장 공식을 반복형태로 바꿔 쓰면, 식 (4-2)과 같다.

$$R_j(0)=0$$
$$R_j(i)=\left(1+\left\lceil\frac{Q_i-J_{i,j}}{G}\right\rceil\right)R_j(i-1)+\left\lceil\frac{q_i-J_{i,j}}{G}\right\rceil \quad \text{for } i=1..M \quad (8)$$
$$PN_j=R_j(M)$$

식 (7)을 이용하여 어드레스 발생기를 설계하는 것보다 식 (8)을 이용하면 PE(Processing Element)를 설계할 수 있다. 일정한 반복이 진행된 후 어드레스를 얻을 수 있기 때문이다. PE를 설계해놓고 이를 반복적 으로 사용하면 임의의 반복회수가 지난뒤에 원하는 결과 즉 주소를 얻을 수 있다.

먼저 스케쥴 다이어그램(sch edule diagram)을 그려보면 어느 PE를 어느 때 사용하 는가를 알 수 있고 중복되는 PE를 설계하면 PE의 숫자를 최소한으로 할 수 있다. PE의 사용은 하드웨어 설계시 용이함을 제공한다. 보다 작은 공간에 어드레스 발생기를 설계할 수 있음을 의미한다. 그림 2는 최소의 PE 수를 표시한다.

여기에서 $J_{M,N}$는 주소 발생 행렬이다. 매핑 공간에서 어느 공간을 연결할 것 인지의 정보가 이 행렬의 원소가 된다.

이미지의 색채는 R, G, B로 3차원이고, 4개의 셀이 반응하게 되므로 3×4의 행렬이 필요하게된다.

이 경우는 최소의 경우이고, 연결이 늘어날수록 행렬의 크기도 증가하게 되며 최대의 연결은 모든 점을 연결하는 3×16의 행렬이된다.

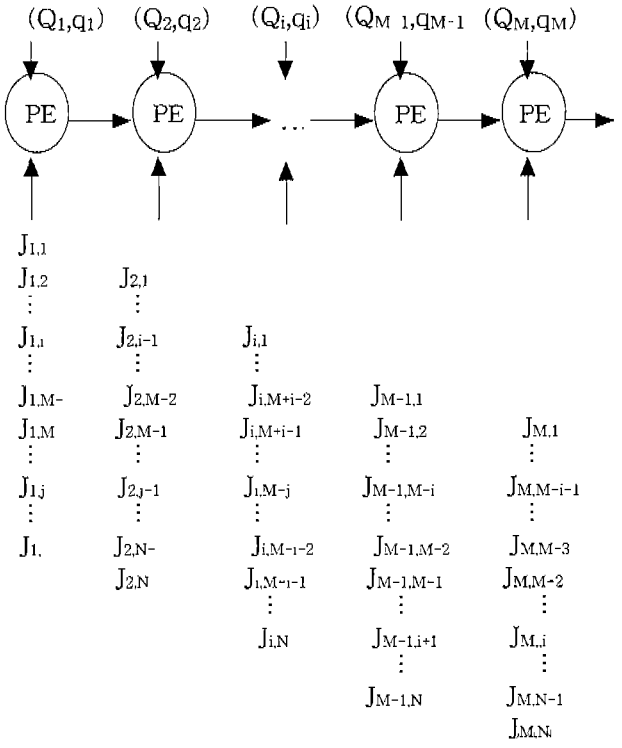

그림 3. 최적화된 시스토릭 어레이
Fig. 3 corresponding systolic array

$$J_{3\times4}=\begin{bmatrix}1&2&3&4\\1&2&3&4\\1&2&3&4\end{bmatrix}$$

$$J_{3\times8}=\begin{bmatrix}1&2&3&4&1&2&3&4\\1&2&3&4&4&3&2&1\\1&2&3&4&1&2&3&4\end{bmatrix} \quad (9)$$

$$J_{3\times12}=\begin{bmatrix}1&2&3&4&2&3&4&1&1&2&3&4\\1&2&3&4&1&2&3&4&1&2&3&4\\1&2&3&4&1&2&3&4&4&1&2&3\end{bmatrix}$$

학습은 수행한 후 에러를 최소화 하기 위해 가중치를 조정함으로써 이루어진다. 에러의 계산은 식 (10)과 같다. 요구되는 출력값과 CMAC 출력사이의 오차는 양의 학습율에 따라 학습하게 된다.

학습에 대한 기본공식은 식 (10)과 같다.

$$y(S)=\frac{1}{C}\sum_{i=1}^{C}W[A_i]$$
$$\Delta W=\beta*[y_d(S)-y(S)] \quad (10)$$

여기서, y(s)는 CMAC 시스템의 출력, C는 입력에 대해 발현하는 셀의 수(Generalization), β는 양의 학습률, $y_d(S)$는 목표값이다.

위 식에 따라서 학습이 이루어지며, 이 과정에서 해싱 충돌은 무시된다. 그러나 정확한 근사화를 요구하는 시스템의 경우 이 충돌이 자주 일어나면 오차가 증가하여 최적의 결과는 얻을 수 없다. 충돌이 발생할 확률은 다음의 적재 밀도가 70%를 초과하면 높아진다. [6]

적재 밀도는 식 (11)을 이용하여 구할 수 있다.

$$\text{적재 밀도}=\frac{\text{association cell 수}}{\text{총 어드레스 수}}=\frac{K}{C} \quad (11)$$

주소의 갯수를 증가시키면 적재밀도가 감소하여 충돌은 감소하겠지만 비 효율적이므로 어소시에이션 셀에 따라서 시스템을 설계하는 것이 바람직하다.

다른 문제로 학습에서 고려해야할 사항은 학습 용량이다. 통상 CMAC의 학습 용량은 메모리의 크기에 의존한다. 엄밀하게 따진다면 용량은 물리 메모리의 크기가 아니고 이용가능한 가중치의 수이다. 이 가중치의 수는 다음과 같이 구할 수 있다.

$$M_A=\frac{(R+C-1)^N}{C^{N-1}} \quad (12)$$

여기서, R은 입력 변수의 범의, C는 입력에 반응하는 어소시에이션 셀 수, N은 입력의 차수이다.

5. PE의 하드웨어 설계

위에서의 주소를 구하게 위한 PE를 설계하면 다음과 같다. 회로는 그림 4와 식 2에 의해서 구현되었다.

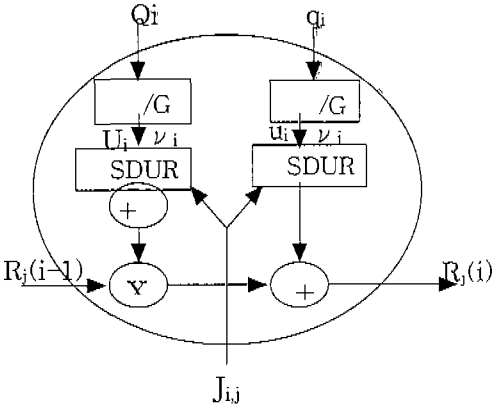

그림 4. PE의 블록도
Fig. 4 Schematic of PE

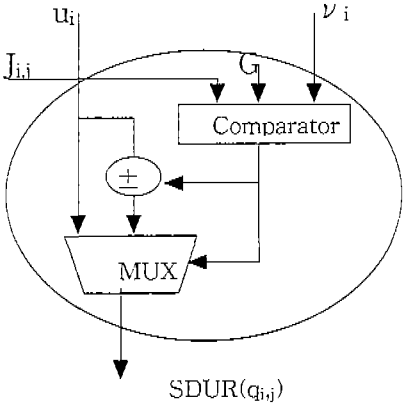

그림 5. SDUR의 블록도
Fig. 5. Schematic of SDUR

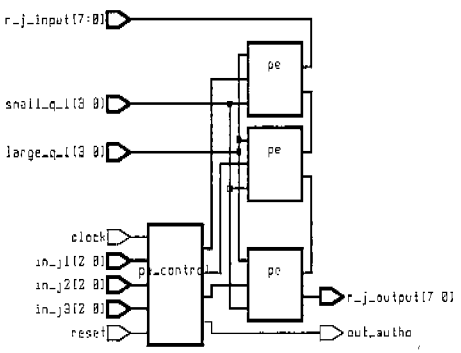

그림 6. 완성된 PE블록
Fig. 6 PE Block Synthesised

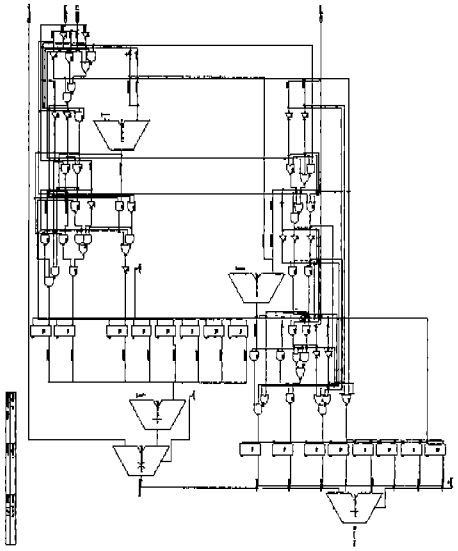

그림 7. PE의 회로도
Fig. 7 Schemetic of PE

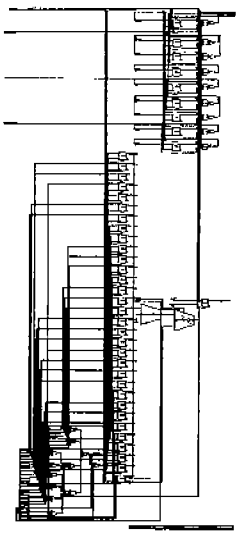

그림 8. PE 제어기의 회로도
Fig. 8 Schemetic of PE controller

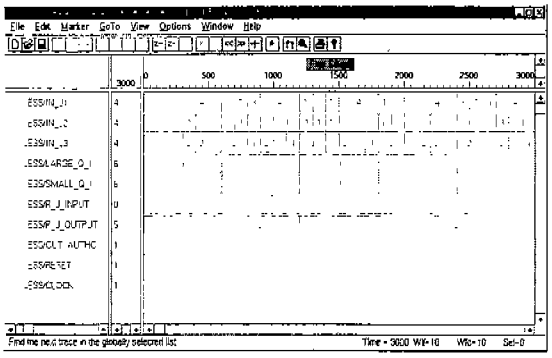

그림 9. PE 블록의 모의실험 결과
Fig. 9 Simulation Result of PE block

6. 결 론

신경망은 여러 비선형 문제와 최적화 문제에서 그 효용성이 인정되고 있는 바 본 논문에서는 CMAC 구조를 색체의 조정에 이용하고자 핵심이 되는 주소를 발생하는 PE를 설계하였고 이를 모의실험하였다. 향후 연구 과제로는 CMAC 매핑의 전단계 B-spline 함수를 퍼지화하여 적용하고, 이를 이용한 새로운 CMAC 구조를 색체 조정에 적용하고자 한다.

참 고 문 헌

[1] 최형진, 정영준, 양해술, 뉴로컴퓨팅, 홍릉과학출판사,pp 248-252, 1994

[2] 김희승, 인공지능과 그 응용, 생능, pp. 333-360, 1994

[3] J. Albus, "A new approach to manipulator control : The cerebellar model articulation controller(CMAC)", Trans. ASME J.Dyn Syst Mear Control,vol 97, pp. 220-227, 1975.

[4] C. H. Chen, Fuzzy Logic And Neural Network Handbook, McGraw-Hill, pp 26.4-26 6,26.7-26.11, 1996.

[5] 김재희, 인공지능의 기법과 응용, 교학사, pp. 286-325, 1990

[6] 이 호, 패턴 처리론, 정익사, pp. 185-197, 1991.

[7] Alfonso F. Cardenas, Data Base Manegement system, Allyn and Bacon, pp 324-327, 1985.

[8] S H Lane et al, "Theory and Development of Higher -Order CMAC Neural Networks", IEEE Control System, pp. 23-30 April 1992

[9] M. Brown and C J Harris, "The Modelling abilities of Binary CMAC", trans IEEE, pp 1335-1339, Novem 19, 1993

[10] P.J Hatley, "Tensor product approximations to data defin ed on rectangular meshs in N-space", The Computer Journal, vol. 19, no. 1, pp.348-351, 1975.

[11] Wang Gang and Liu Yongxiang and Cao Hongjia, "A Nonlinear Interpolation Approach for Image Process", Proc. of ICSP'98, pp. 1060-1063, 1998.