한국 퍼지 및 지능시스템학회 논문지 2000, Vol. 10, No. 5 (2000)466~477

운반차-막대 시스템을 위한 적응비평학습에 의한 CMAC 제어계

CMAC Controller with Adaptive Critic Learning for Cart-Pole System

권 성 규

Sunggyu Kwon

계명대학교 기계공학과

요 약

이 논문에서는 운반차-막대 시스템을 제어하기 위한 CMAC을 이용한 적응 학습 제어계를 개발하기 위하여, 적응비평학습을 이용하는 신경망 제어계에 관한 여러 연구 문헌들을 조사하고, ASE 요소를 이용하는 적응비평학습 기법을 CMAC을 바탕으로 하는 제어계에 통합하였다. 적응비평학습 기법을 CMAC에 구현하는데 있어서의 변환 문제를 검토하고, CMAC 제어계와 ASE 제어계가 운반차-막대 문제를 학습하는 속도를 비교하여, CMAC 제어계의 학습 속도가 빠르기는 하지만, 입력 공간의 더 넓은 영역에 대해서는 학습효과를 발휘하지 못하는 문제의 관점에서 적응비평학습 방법이 CMAC의 특성과 어울리는지를 고찰하였다.

ABSTRACT

For developing a CMAC-based adaptive critic learning system to control the cart-pole system, various papers including neural network based learning control schemes as well as an adaptive critic learning algorithm with Adaptive Search Element are reviewed and the adaptive critic learning algorithm for the ASE is integrated into a CMAC controller. Also, quantization problems involved in integrating CMAC into ASE system are studied. By comparing the learning speed of the CMAC system with that of the ASE system and by considering the learning generalization of the CMAC system with the adaptive critic learning, the applicability of the adaptive critic learning algorithm to CMAC is discussed.

1. 서 론

운반차-막대(cart-pole) 또는 도립 진자(inverted pendulum) 시스템 문제는 지능 제어계의 성능을 검증하기 위해서 널리 채택되는 문제로써, 수평 직선을 따라 움직이는 운반차에 세워져 중심을 잃은 막대를 넘어지지 않도록 운반차에 적절한 힘을 가하는 제어계(broom-balancer[1])를 개발하는 것이다. 이 시스템을 제어하는 문제를 막대 균형잡기(pole-balancing, broom-balancing[2]) 문제로 부르기도 한다. 이 시스템은, 구조는 단순하지만 불안정한 제어 대상 시스템의 대표적인 것으로서, 우주선 로켓이 발사될 때, 로켓 추진장치가 로켓 추진 벡터의 꼭지점에 관해서 균형을 유지하여야 하는, 추진장치의 동적 모델을 나타내는 다소 인공적인 시스템으로 간주할 수 있다.

운반차-막대 시스템의 제어 문제는 지난 35여 년간 고전적인 여러 방법들로 광범위하게 연구되었다. Bang-bang 유형의 제어를 비롯하여[3], observer-regulator 유형의 동적 안정기(stabilizer)를 사용해서 이중 도립막대를 안정화하는 제어계도 있으며[4], 아래로 처져 자유 평형 상태에 있는 진자를 흔들어 도립위치까지 추켜세울 수 있는 제어 시스템도 제시되었다[5]. 아주 최근에, 트랙 상의 운반차의 위치도 제어하면서 운반차에 세워진 막대의 균형을 잡기 위하여, 상태 공간과 주파수 응답 방법들의 혼합을 사용하는 강인한 디지털 제어기를 설계하여, 비선형 마찰로 인한 극한 사이클 요동을 효과적으로 줄일 수 있었던 예도 있다[6].

비록 다양한 고전적인 제어 기법들이 있다하더라도, 그들은 실질적으로는 거의 같다. 먼저 시스템의 동적 행태를 기술하는 미분 방정식을 세워 운반차-막대 시스템에 대한 수학 모델을 만들고 나서 안정성, 제어성 및 관측성 요구 조건들을 만족하는 제어 규칙과 상태 되먹임 변수 값들을 구한다. 그러기 위해서는 시스템의 상태를 관찰하는 모델도 갖추어야 하며, 시스템을 수학적으로 정확하게 기술하여야 하는데, 시스템의 물리적 매개변수들과 그것의 동적 특성들에 관한 지식이 부족하다면, 고전적인 제어 이론을 사용하는 해석적 풀이 방법들을 적용하는 것은 무리이다. 그래서, 최근에는 적응 또는 학습제어 방법들을 많이 사용한다.

1963년에 운반차-막대 시스템을 제어하기 위한 인

공적인 신경 요소를 이용한 제어기가 개발되었지만[2], 아직도, 운반차-막대 시스템을 제어하기 위해 신경망을 훈련하는 것은 어려운 문제로 간주되고 있으며 신경망 학습 알고리즘들을 시험해보는 인기 있는 대상 문제이다.

특히, 이 문제가 적응 네트워크 연구에서 많이 다루어지는데는 몇 가지 이유가 있다[1,7]. 첫째로, 문제를 설명하기가 간단하고 이해하기도 쉽지만, 제어의 관점에서는 꽤 복잡한 문제이다. 우리들이야 조금만 연습하면 손바닥에 세운 빗자루나 긴 막대의 균형을 쉽게 잡을 수 있지만, 이 때 손이 3차원 공간에서 자유로이 움직일 수 있음을 간과해서는 안된다. 그래서, 예를 들어, 손을 수평면 내에서만 움직여야 한다거나, 손을 수평면 상의 일직선을 따라서만 움직이도록 한다면, 누구라도 그런 운반차-막대 시스템을 제어하기는 쉽지 않을 것이다. 두 번째로, 운반차-막대 문제는 본질적으로 불안정한 제어 시스템의 교과서적인 예제이다. 그래서, 전통적인 제어 이론으로 문제가 광범위하고 상세하게 분석되어 있어서, 신경망 제어기들을 평가하는데 편리하게 적용할 기준이 마련되어 있는 셈이다. 세 번째로, 문제의 시스템은 단순하여서 계산적으로 다루기 쉬울 뿐만 아니라 큰 힘들이지 않고 실험 장치를 갖출 수 있다. 넷 째로, 시스템의 제어 문제는 실시간 문제이기 때문에 제어기들의 응답시간에 대한 제약들이 있다. 그래서, 컴퓨터 시뮬레이션에서는 작동하나 실시간으로 구현하지 못하는 너무 느린 제어기들은 기껏해야 학구적 관심이 될 뿐이다. 마지막으로, 이 문제는 많은 다른 제어 문제들의 전형이므로, 이 문제를 통해서 적응 네트워크들을 어떻게 사용할 지에 대한 방안을 강구한 다음 다른 제어 문제들을 해결하는데 활용할 수 있을 것이다.

신경망은 근본적으로 classification 네트워크이고 신경망에서의 classification은 비선형 변환의 한 유형이다. 그래서 신경망이 복잡한 변환들을 즉흥적으로 학습하는 능력이 신경망 응용의 가장 두드러지는 특성이다. 이 특성의 피상적인 이점은 우리가 일일이 닫힌 형태의 해석적인 함수를 유도하지 않아도 된다는 것이다. 그러나, 더욱 중요한 것은, 신경망이 수학적으로 다루기 힘든 변환들을 학습할 수 있다는 것이다[8].

수학적으로 다루기 힘든 변환들을 신경망이 학습하도록 하기 위해서는 신경망을 가르치는 교사가 있어야 한다. 신경망은 교사에 의한 훈련과 학습 평가 과정을 거치면서 어떤 유형의 고전적인 제어를 모방과 배움을 통해서 학습한다. 그러나 기존의 제어계로부터 배운다는 것은 쓸모가 없는 것이, 일반적으로, 학습에 의한 신경망의 제어 역량이 준거로 삼은 기존의 제어계의 성능을 능가하기를 기대할 수 없기 때문이다. 그래서, 많은 신경망들은 해석적 모델과 상관없는 제어기, 즉, 인간 교사로부터 배운다. 아마도 신경망의 가장 강력한 특성은 제어된 시스템 자체를 모델 하는 능력이다[8].

신경망을 응용한 첫 사례이자 제어에 적용한 첫 사례는, 1963년에 어떤 불안정한 시스템을 최적 제어하는데 있어서 "ADALINE(ADAptive LINEar element)"이라는 뉴우런의 역량을 입증할 목적으로 제작된 adaptive broom balancer이다. 이 장치에서는 bang-bang 방식의 제어를 사용했는데, 시스템을 수동으로 운전하는 인간 운전자를 관찰함으로써 단 하나의 ADALINE을 훈련하였다[9]. 또 다른 연구에서, 운반차-막대 시스템과 ADALINE을 컴퓨터로 시뮬레이션하고 네트워크에 시스템 상태를 나타내는 시각 정보를 입력하였을 때, 틀린 응답이 없지는 않았지만 네트워크는 진자가 넘어지는 것을 성공적으로 막을 수 있었다[1].

운반차-막대 시스템의 제어를 위한 적응 문제 해결 시스템 중에는 "boxes system(상자시스템)"이 있다. 이 제어계에서는 시스템을 정의하는 4-차원 상태 공간을 구성하는 네 개의 연속적인 상태 변수들을 quantizing함으로써 상태 공간을 서로 독립적인 영역들로 구분하고 그 영역을 "box(상자)"라 불렀다. 그렇게 해서 문제를 몇 개의 독립적인 하부 문제들로 분할하고 각 하부 문제들에 대해서 동일한 규칙을 사용하여 문제의 해결책을 학습하는 시스템을 제시하였다[10]. 또한, 상자시스템을 "ASE(Associative Search Element)"라는 뉴우런과 같은 제어 요소들로 구현하고, "ACE(Adaptive Critic Element)"라는 적응력이 있는 요소를 부가하여, 상자시스템에 의한 학습 제어 시스템과, ASE와 ACE로 구성된 학습 제어 시스템을 비교함으로써, 운반차-막대 시스템에 관한 적응 제어 기법에 큰 공적을 남긴 연구 결과도 있다[11].

퍼지 기법을 이 문제에 적용하는 연구 사례도 많이 보고되고 있다. 대부분의 퍼지 논리 함수들을 구현하는데 사용할 수 있고 하드웨어로 구현하기가 꽤 용이한 학습 퍼지 네트워크를 개발하여 운반차-막대 시스템의 제어에 적용한 예[12], 퍼지 제어기의 계수들을 정하기 위하여 유전 알고리즘의 탐색 역량을 활용하는 제어 기법을 운반차-막대 문제에 적용한 예 등이 있다[13]. 또한, 시스템의 시각 정보를 되먹임 받는 퍼지-논리 제어기 실험 장치로 제한적인 시간 동안이나마 막대의 균형을 유지할 수 있는 운반차-막대 제어 시스템도 있다[14].

CMAC을 이용하여 이 문제를 위한 제어계를 구축

하는 연구 예도 많이 있다. 상자시스템을 이용하는 방법과 신경망을 이용하는 방법들을 비교한 연구에서, CMAC을 이용하는 제어 계략이 가장 빠른 학습 속도를 보였다[15]. 또한, 평면상에서 운동하는 운반차에 직교하는 두 힘을 작용하여, 운반차에 서있는 막대의 균형잡는 문제를 위한 적응비평학습 제어계에 대해서는 8개의 상태 변수를 갖는 CMAC이 이용되었다[16]. 상자시스템을 이용하는 방법에 기본을 두되, 연속적인 입력 공간을 불연속적인 공간으로 변환하는데 있어서 CMAC의 중간변수 개수만큼의 정보의 채널을 설정함으로써, ASE와 ACE에 의한 제어 계략보다 더 빠르고 더 효율적인 학습을 달성한 예가 있으며[17], ASE와 ACE에 의한 제어계와 적응비평학습 방법을 이용하여, CMAC을 주요 학습 요소로 하는 제어계를 시뮬레이션한 예에서는, 제어계가 비교적 불안정한 초기 상태의 운반차-막대 시스템의 균형을 잡는 역량을 성공적으로 학습하였다[18]. CMAC을 주요 제어 요소로 이용하는 적응비평학습에 관한 이전의 연구[15,16]에서 적절한 선정 기준 없이 학습 매개변수 값들을 정하였는데, [19]에서는 CMAC의 메모리 규모와 구조를 결정하기 위한 방법론을 검토하고 CMAC에 바탕을 둔 적응비평학습에서 사용할 학습 매개변수 값들을 선정하는 지침을 마련하였다.

[7]에서는 운반차-막대 문제를 철저하게 분석하고, 앞서 있었던 연구 결과들에 함축되어 있거나 잘못된 개념들을 정리하여, 이 문제를 위한 신경망 훈련 알고리즘들을 개발하는데 있어서 유용하게 이용할 수 있도록 잘 정의된 규준이 될 만한 조건들을 제안하였다.

이 논문에서는 운반차-막대 시스템을 제어하기 위한 CMAC을 이용한 적응 학습 제어계를 개발하기 위하여, 같은 시스템에 대한 다양한, 특히 지능 및 신경망을 이용한 여러 연구 문헌들을 조사하고, ASE 요소를 이용하는 적응비평학습 기법을 CMAC을 바탕으로 하는 제어계에 통합하는 방안을 연구하였다. 이를 위해, 연속적인 입력 변수들을 CMAC에서 불연속화하는 것과 관련된 문제들을 고찰하였으며, CMAC 제어계와 ASE 제어계가 운반차-막대 문제를 학습하는 성능을 비교함으로써 CMAC의 특성이 과연 적응비평학습과 잘 어울리는지에 대해서 검토하였다.

2. 운반차 - 막대 시스템 문제

운반차-막대 시스템은 긴 막대기나 자루가 긴 비의 손잡이 끝을 손가락 끝이나 손바닥에 세워서 균형을 잡는 문제의 특별한 형태이다. 가장 널리 사용되는 운반차-막대 시스템은 [11]에 기술된 것이다. 막대기의 끝이 운반차 중심에 회전 조인트로 피벗 되어 세워져 있고 막대기는 운반차의 운동 방향에 평행한 수직 평면에서만 흔들릴 수 있다. 막대기의 기울기가 미리 설정된 한도를 넘거나 운반차가 트랙의 한계를 벗어나 경계와 부딪치지 않으면서, 서 있는 막대기를 넘어지지 않게 균형을 잡기 위해서는 제한된 길이의 트랙에서 운반차에 적당한 힘을 가할 수 있어야 한다.

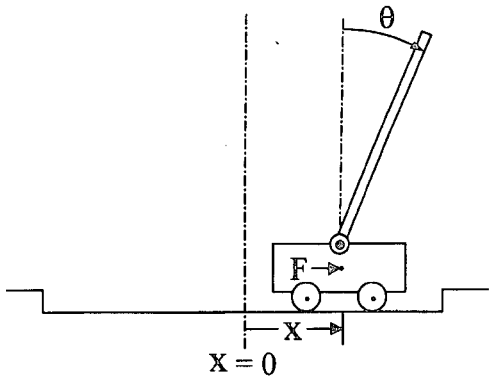

그림 1. 운반차-막대 시스템의 개략도[11]

그림 1은 일직선 트랙을 따라 운동하는 바퀴가 달린 운반차에 베어링 피벗으로 막대가 장착되어 있는 운반차-막대 시스템을 보여준다. 운반차-막대 시스템의 동역학은 네 개의 변수(트랙상의 운반차의 위치 x, 운반차의 속도 $\dot{x}$, 수직 축에 관한 막대의 각도 θ, 및 막대의 각속도 $\dot{\theta}$)를 써서 다음과 같은 비선형 미분방정식들로 기술할 수 있다[17]:

$$\ddot{\theta}_t = \frac{g\sin\theta_t + \cos\theta_t\left[\dfrac{-F_t - ml\dot{\theta}_t^2\sin\theta_t + \mu_c sgn(\dot{x}_t)}{m_c+m}\right] - \dfrac{\mu_p\dot{\theta}_t}{ml}}{l\left[\dfrac{4}{3} - \dfrac{m\cos^2\theta_t}{m_c+m}\right]} \quad (1)$$

$$\ddot{x}_t = \frac{F_t + ml[\dot{\theta}_t^2\sin\theta_t - \ddot{\theta}_t\cos\theta_t] - \mu_c\mathrm{sgn}(\dot{x}_t)}{m_c+m} \quad (2)$$

여기서 g = -9.8 m/s^2 중력가속도, m_c=1.0 kg 운반차의 질량, m = 0.1 kg 막대의 질량, l = 0.5 m 막대 길이의 반, μ_c = 0.0005 트랙에서 운반차의 마찰계수, μ_p = 0.000002 운반차에서 막대의 마찰계수, F_t = ±10.0 Newtons 시각 t에 운반차의 질량중심에 가해진 힘이다.

제어 힘의 크기를 제외하고는, 다른 매개변수들을 다소 달리 선정하더라도 제어 문제의 본질에는 거의 영향을 미치지 않는다. 운반차를 무겁게 하고 막대는 가벼운 것으로 하면, 운동 방정식 (1)이나 (2)에서 더 복잡한 비선형 항들의 영향이 줄어든다. 시뮬레이션 결과들에 의하면, 운반차의 질량이 커짐에 따라 시스

템을 제어하는데 필요한 힘의 크기가 비례해서 증가하는 것 외에, 문제의 해에 거의 영향을 미치지 않는 것으로 나타난다[7].

[11]에서 사용한 운동방정식들에는 작은 마찰 항들이 포함되어 있는데, 작은 값의 마찰계수들은 문제를 수식화하는데 있어서의 단지 약간의 현실감을 더하는 겉치레 외에 별 영향은 미치지 않는다. 그러나, 제어기들이 운반차가 느리게 멀어지거나 요동하게 하는 경우들에서 마찰은 운반차가 정지하는데 도움이 될 것이다.

운반차-막대 시스템에 관한 많은 연구들에서, 운반차에는 일정한 크기의 힘이 가해진다. 그래서, 제어계의 최종 역할은 단지 운반차에 가하는 힘의 방향을 정하는 일 뿐이다. 이 제어 계략이 이른바 bang-bang 제어이다. 제어하는 힘의 크기를 연속적으로 변화할 수 있다면 시스템을 더 잘 제어할 수 있지만 그때에도 실제로 어떤 모터가 생성할 수 있는 힘의 크기에는 한계가 있음을 명심해야 한다. 어떤 제어기의 성능은 운반차-막대 시스템을 안정화할 수 있는 초기 조건들의 범위, 운반차를 트랙의 중앙에 위치하게 하는데 걸리는 시간 및 시스템이 안정화되고 난 뒤에 잔존하는 요동이나 변위의 범위로 산정할 수 있다. 상당히 오랜 시간 동안 지속적으로 막대의 균형을 유지하지 못하는 제어기들에 대해서는 그 때까지의 시간으로 제어기의 성능을 산정하기도 한다[7].

3. ASE에 의한 적응비평학습

이 절에서는 [11]의 내용을 요약해서 기술한다. 그림 1의 운반차-막대 시스템에 대해서, 운반차는 1차원 트랙의 좌우 경계 내에서 자유로이 움직일 수 있으며, 막대는 단지 운반차와 트랙의 수직 평면에서만 자유로이 움직일 수 있고, 제어기는 단지 일정한 크기의 힘 F를 운반차의 중심에 "왼쪽"이나 "오른쪽"으로 가할 수 있다.

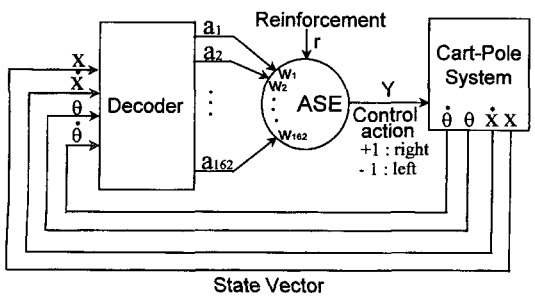

그림 2. 운반차-막대 시스템을 위한 ASE 제어계의 개략도[11]

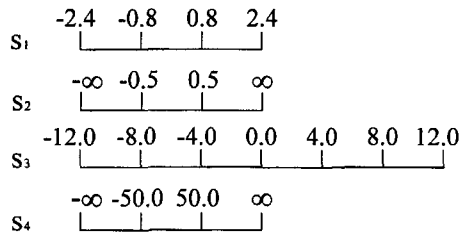

그림 3. Quantization thresholds에 의한 상태변수들의 불연속화

그림 2에서 decoder는 운반차-막대 시스템의 상태벡터, $\mathbf{S}=(s_1, s_2, s_3, s_4)$를 접수한다. 이 벡터의 성분들 중에, s_1은 트랙에서 운반차의 위치 x, s_2는 막대가 수직선과 이루는 각도 θ, s_3는 트랙을 따라 운동하는 운반차의 속도 $\dot{x}$를 나타내고 s_4는 막대가 피벗에 대해 회전하는 각속도 $\dot{\theta}$를 나타낸다.

[11]에서는 이 시스템을 제어하기 위해, 위의 네 개의 연속적인 변수들로 구성된 4-차원 상태 공간을 네 개의 상태 변수들을 quantizing함으로써 서로 별개의 영역인 상자들로 구분하였다. 예를 들면 다음과 같은 quantization thresholds[17]에 의해 연속적인 입력변수들을 그림 3에서 보는 것처럼 여러 개의 마디(knot)[20]로 분할하여 불연속화한다.

1) x : ± 0.8, ± 2.4 [m]
2) $\dot{x}$: ± 0.5, $\pm\infty$ [m/sec]
3) θ : 0, ± 4.0, ± 8.0, ± 12.0 [deg]
4) $\dot{\theta}$: ± 50.0 $\pm\infty$ [deg/sec] (3)

이렇게 되면, 무수히 많은 실수 변수 값들로 측정될 수 있는 원래의 연속적인 입력변수는 단지 몇 개의 변수 값들만을 가지도록 불연속화된다.

위와 같이 quantize하면, 네 개의 변수들은 그림 3에서 보는바와 같이, s_1, s_2, 및 s_4는 각각 세 개씩, s_3는 여섯 개의 마디로 나누어져서 불연속화된다. 그래서, 예를 들면, $-2.4 \le x < 0.8$ 구간에 해당하는 x의 모든 변수 값들은 단 하나의 동일한 변수 값으로 표시된다. 그래서, 연속적인 입력변수 s_i가 ${}^iN$개의 구간으로 나누어져서, ${}^1N=3$, ${}^2N=3$, ${}^3N=6$, ${}^4N=3$. 즉 s_1에 대해서는 단지 세 개의 입력변수 값만 존재한다는 것이다. 그러므로, 불연속 상태공간에서는 입력변수의 값을 수치 값으로 나타내기보다는 그 구간을 표시하는 이름표를 붙이는 것이 편리하다.

그래서, 입력변수 s_i를 ${}^iN$개의 구간으로 분할하였을 때, j번째 구간에 ${}^im_j (j=1, 2, \cdots, {}^iN)$이라고 이름표를 붙이기로 하자. 그러면, 그림 3에서 s_1의 [-2.4, -0.8) 구간에는 ${}^1m_1$를, 또 s_3의 [8.0, 12.0]구간에는 ${}^3m_6$라는 이름표를 붙인다. 그러면, 시스템의 상태는 그 시각에

개별 입력변수들을 나타내는 어떤 구간들의 이름표들의 조합으로 표시되어, 예를 들어, $S^* = (0.9, 42.7, -10.5, -0.1)$는, 이제, $M_l = ({}^1m_3, {}^2m_3, {}^3m_1, {}^4m_2)$로 표시된다. 그래서, 시스템의 상태는 ${}^1N = 3$, ${}^2N = 3$, ${}^3N = 6$, ${}^4N = 3$ 등의 경우의 수 조합에 따라, 단지 $162 (= {}^1N \times {}^2N \times {}^3N \times {}^4N)$ 가지의 이름표 조합들로 나타내진다. 그러므로, 위의 M_l과 같은 시스템의 상태가 162가지가 있다는 의미이고, $l = 1, 2, \cdots, 162$.

이제 다음과 같이 162개의 성분을 가지는 어떤 벡터 A를 정의하자:

$A_l = (a_1, a_2, \cdots, a_k, \cdots, a_{162})$. 성분 a_k는 2진 값을 갖는 변수로써, 다음과 같은 관계에 의해 그 값이 0 아니면 1로 정해진다:

$$a_k = \begin{cases} 1 & for \quad k = 1 \\ 0 & otherwise \end{cases} \tag{4}$$

$$l = \{f({}^1m_{1j}) - 1\} \times 54 + \{f({}^2m_{2j}) - 1\} \times 18 + \{f({}^3m_{3j}) - 1\} \times 3 + f({}^4m_{4j}) \tag{5}$$

$$ij = f({}^im_{ij}) \tag{6}$$

여기서, f는 독립변수 ${}^im_{ij}$의 ij를 그 값으로 주는 함수이다. 예를 들어, $3 = f({}^im_3)$. 시스템의 아무 상태 벡터 S^*에 대해서 오직 하나의 2진 변수만이 그 값이 1로 표기되고 나머지는 모두 0이므로, 만약 $l = 3$이면, $A_3 = (0, 0, 1, \cdots, 0, 0)$로 표시되는 것을 의미한다. 그러므로, 이제 시스템의 상태 S^*는 A_l로 표시된다.

벡터 A를 구성하는 아무 성분 a_k는 오직 0 아니면 1을 나타내기 때문에 각 성분은 2진 정보를 저장하는 채널로 간주하자. 그러면 벡터 A는 162개의 채널로 구성되는 정보의 통로로 볼 수 있고, 이 통로는 아무 때라도 단지 한 채널만 켜진 상태(즉 특정 채널에 해당하는 성분의 값이 1)이고 나머지는 모두 꺼져 있는(즉 채널에 해당하는 성분의 값이 0) 통로이다. 몇 번째 채널이 켜져 있는지는 식 (4)와 (5)에 의해 알 수 있다.

그림 2에서 decoder는 특정 시각에 시스템 상태 벡터 S^*를 되먹임 받아서, 식 (4)에 의해, 벡터 A_l을 출력한다. ASE에는 decoder의 채널과 1대 1로 연결되어 있는 메모리가 있고, 그 메모리에는 decoder로부터 연결된 채널에 실려있는 값에 곱해지는 가중치 $w_k(t)$(:ASE의 매개변수 벡터)가 저장되어 있다. ASE는 decoder와 연결된 채널들에 이 개별 가중치들을 곱하여 합한 값을 출력으로 내놓는다. Decoder의 출력인 벡터 A_l는 단 하나의 성분 a_l만이 값이 1이고 나머지 성분들의 값은 모두 0이므로, ASE의 출력에 반영되는 가중치는 162개중에서 그 값이 1로 표기된 채널과 연결된 단 한 개뿐이다. 이 가중치들에 의해 ASE의 출력이 정해지고, 이 출력에 따라서 운반차에 가해지는 일정한 크기의 힘의 방향이 결정된다.

ASE의 출력 $y(t)$는 입력 벡터 $A(t)$로부터 다음과 같이 결정된다:

$$y(t) = \sum_{k=1}^{n} w_k(t) a_k(t) + \text{noise}(t) \tag{7}$$

여기서 noise(t)는 실수 값을 갖는 확률(random) 변수이다[17]. 힘의 방향을 정하는 플랜트의 출력은 다음과 같이 결정된다:

$$Y(t) = f[y(t)]$$

$$Y(t) = f[\sum_{k=1}^{n} w_k(t) a_k(t) + \text{noise}(t)] \tag{8}$$

여기서 f는 다음과 같은 함수이다:

$$f[y(t)] = \begin{cases} +1 \text{ 만약 } y(t) \geq 0 \text{(오른쪽으로 힘을 작용)} \\ -1 \text{ 만약 } y(t) < 0 \text{(왼쪽으로 힘을 작용)} \end{cases} \tag{9}$$

식 (7)에서, $a_k(t)$는 $k = 1$인 경우에 그 값이 1이고 그 외에는 값이 모두 0이기 때문에, noise(t)를 무시한다면, $y(t) = w_1$로 볼 수 있다.

이제, ASE의 메모리 요소인 가중치 $\omega_k(t)$들을 어떻게 생성하고 갱신하는지 살펴보자. 제어계가 훈련된 뒤, 운반차가 트랙의 중앙에 있고 막대가 균형이 잡혀서 있을 때는, 시스템은 아무 힘도 실제로 필요하지 않다. 왼쪽이나 오른쪽으로 제어 조처들이 있을 가능성은 반반으로 서로 상쇄될 것이다. 만약 시스템에 오른쪽으로 힘을 가해야 한다면, 플랜트는 왼쪽보다는 오른쪽으로 힘을 작용하게 되는 조치를 취해야 할 것이며, 그 역도 마찬가지다. 어떤 시각 t에 시스템의 상태가 k번째 채널에 의해 나타내질 때, 가중치 $w_k(t)$가 양의 값이면, 플랜트의 출력이 힘을 오른쪽으로 작용하게 하는 결정을 내리게 되므로, 가중치들은 단지 제어 조처 그 자체보다는 제어 조처의 확률을 결정하는 것이다.

가중치 $w_k(t)$는 다음 규칙에 따라 매 시각 변한다:

$$w_k(t+1) = w_k(t) + a \cdot r(t) \cdot e_k(t), \quad k = 1, 2, \cdots, 162. \tag{10}$$

여기서 α는 가중치 $w_k(t)$의 변화율을 결정하는 양의 상수이고, $r(t)$는 시각 t에서의 강화(reinforcement)를 나타내는 실수값 상벌 신호로, 0 또는 -1이며, $e_k(t)$는 시각 t에 입력 통로 $a_k(t)$의 적격성(eligibility)이다.

$r(t)$는 막대기의 기울기가 미리 설정된 한도를 넘어

서지 않고 운반차가 트랙의 한계를 벗어나 경계와 부딪치지 않는 동안에 0으로 남아있고, 그렇지 못하는 실패의 경우에는 -1이 된다. 그래서, 이 모델에 대해서, ASE의 가중치들은 시스템이 실패하지 않는 한, 갱신되지 않는다. 그러다가, 시스템이 실패하면 $r(t)$ = -1이 되고, 모든 가중치들이 각각의 적격성에 따라 어떤 양만큼 줄어들 것이다. 경로의 적격성이 클 수록 그 채널은 실패 사건의 발생에 대한 책임이 더 커져서, 그것의 가중치는 더 큰 징계를 받아야 한다. 식 (10)은 어떤 채널 k에 입력이 있을 때는 언제나 적용되는 가중치 갱신 규칙인데, 이 규칙과 위에서 기술한 $r(t)$에 대한 조건에 따라 채널 k와 연관된 가중치가 갱신된다.

적격성 $e_k(t)$는 다음과 같은 식에 따라 쇠퇴하며, 여기서 $\delta(0 \le \delta \le 1)$는 추적 쇠퇴율을 결정한다.

$$e_k(t+1) = \delta e_k(t) + (1-\delta)Y(t)a_k(t),\ k = 1,\ 2,\ \cdots,\ 162. \tag{11}$$

높은 δ는 낮은 적격성 쇠퇴율로 귀착되고 적격성의 낮은 증가는 어떤 입력이 채널 k를 지나갈 때 존재한다. 입력을 갖는 채널의 가중치는 입력이 없는 채널들만큼 많이 변화하지 않는다.

채널 k의 값 $a_k(t)$가 0 아니면 1이기 때문에, k=1인 채널에만 의미 있는 신호가 실려 있어서, $a_1(t)$ = 1. 적격성은 쇠퇴하기도 하지만, $(1-\delta)Y(t)a_k(t)$만큼 증가도 한다. 입력이 실리지 않은 채널의 적격성은 인자 δ만큼 감소한다. 적격성은 그래서 부호가 있는 곱 $Y(t)a_k(t)$의 추적이다. 만약 $Y(t)$와 $a_k(t)$ 둘 다가 음이거나 양이면, 공적이 인정되고, 그들이 반대 부호들을 가지면, 징계가 내려진다. 그래서 추적은 얼마나 오래 전에 어느 채널로 입력이 전달되었는지 뿐만 아니라 그 채널로 입력이 전달되었을 때 취해졌던 결정에 관한 정보도 담고 있다.

지금까지 [11]에서 기술한 적응비평학습 과정을 간략하게 기술하면 다음과 같다: 학습을 시작할 때, 제어계는 학습된 바가 없으므로 ASE의 메모리 가중치의 값이 모두 0이고, 물론 적격성 값들도 모두 초기화된 0의 값들을 가지고 있어서, 모든 채널에 대해서, $w_k(0) = 0$, $e_k(0) = 0$. 시스템이 학습 대상 상태 S = (s_1, s_2, s_3, s_4)에서 해제되고, 첫 시각에서 ASE의 가중치의 합이 0이므로(식 (7)와 식 (8)), 힘은 오른쪽으로 가해지고(식 (9)) 그때 시스템의 상태가 변한다. 달라진 시스템의 상태 변수들이 다음 시각에 decoder로 되먹임된다. 달라진 시스템의 상태가 정해진 기준에 대해서 균형이 유지되는지 아니면 실패로 선언되는가에 따라, 가중치가 갱신되고(식 (10)) 다음 학습을 시도한다. 제어계에는 운반차-막대 시스템의 모델이 포함되지 않아서, 제어계는 시스템의 동역학은 전혀 모른다. 단지 시스템의 상태 변수 값들만 필요로 할뿐이다. 그러므로, 실험 장치에서는 제어계를 위해서 단지 상태 변수 값들만 측정하면 된다. 이 때, 적격성 값들은 다시 초기화되고(식 (11)), 가중치들은 앞선 학습에서 경신된 내용이 다음 학습으로 이어진다. 학습의 다른 시각마다 적격성 값들은 매번 초기화된다. 이 과정을 반복해서 시스템이 100,000번의 시각이 지나는 동안에도 계속해서 균형을 유지하면, 학습 대상으로 삼았던 시스템 상태에 대해서 제어계가 1회의 성공적인 학습을 완료했다고 본다. 같은 학습 시작 시스템 상태에 대해서, 그런 완성된 학습이 100번 이상이 지속되면, 더 이상의 학습은 무의미한 것으로 보고 제어계는 학습을 종료한다.

ASE 제어계에서, decoder는 아무 시스템 상태에 대해서 162개의 채널 중에서 단 하나의 채널만이 시스템의 상태를 표시한다. 그래서, 실패가 없을 때는, 162개의 채널들 중에 단지 한 채널만이 제어 조처에 대한 책임이 있어, 그 채널에 연결된 가중치만 변화된다. 이 극히 제한적인 학습 정보 저장 계략의 결과로서, 특정한 채널 외의 다른 채널에 대해서는 학습의 아무 파급 효과도 일으키지 않는다.

다른 한편으로, [10]의 상자시스템에서 상태 공간을 quantize하고, 메모리 요소와 상자들에 의한 메모리 위치 선정 등에 의해 순람표(look-up table)를 이용하여 제어 조처를 취하는 방식을 주목할 필요가 있다.

이러한 상자시스템의 특성을 살펴보면, CMAC을 이 문제에 활용할 수 있는 여지가 있다. 특히, CMAC이 가지는 학습일반화(learning generalization)의 특성과 메모리 관리 특성[21]은 CMAC을 이용한 제어계가 좋은 특성을 보일 것으로 기대할 수 있게 한다. 운반차-막대 시스템을 지배하는 운동방정식에 의한 계의 행태는 연속적이고 미분 가능이다. 그래서, 만약 단 하나의 채널만이 시스템의 어떤 특정 상태를 나타낸다기 보다는, 그 채널과 이웃하고 있는 몇 개의 채널들이 더불어 시스템의 상태를 나타낸다면, 학습 정보가 복수의 메모리에 반영될 수 있고 그렇게 되면 학습 속도는 더 빠를 것이다. 이것을 달성하기 위한 방법으로는, CMAC을 위의 ASE에 의한 제어계에 통합하는 것이다.

4. CMAC을 이용한 적응비평학습

운반차-막대 균형잡기 문제에 대해서, 연속적인 입력변수들로 정의되는 시스템의 상태는 벡터 [x, $\dot{x}$, θ,

$\dot{\theta}$로 나타낸다. 이 상태를 3절에서 설명한 벡터 A로 나타내면, 162개의 성분 중에 단 하나의 성분에 의해 시스템의 상태가 표시된다. 그렇지만, 모든 입력변수가 연속적이고 그 변수 값의 작은 변화에 대해 시스템의 행태가 별 차이 없이 부드럽게 변한다면, 시스템의 행태는 상태 벡터의 작은 변화에 대해서도 연속적이고 부드럽게 변하는 것으로 기대할 수 있다. 입력변수 s_i의 어떤 실수 값이 시스템의 아무 상태를 정의하는 성분 값이면, 그 실수 값과 가까운 어떤 범위 내의 실수 값들은 어떤 특정한 시스템의 상태와 비슷한 상태를 정의하는 성분 값으로 이해한다. 예를 들어, s_3 = -0.5에 대한 시스템의 상태는 s_3 = -0.6이나 s_3 = -0.4에 대한 상태와 비슷하여, 이들 세 입력변수 값에 대한 시스템의 출력에는 큰 차이가 없을 것으로 기대하는 것이다. 이것은 입력상태공간의 이웃 개념 [21]에 바탕을 둔 것이다.

그래서, 연속적인 어떤 입력변수를 불연속화하여 분할하고 그 분할된 변수들에 이웃 개념을 적용하기 위해서는, 애초에 불연속화할 때 분할 구간의 크기와 그 이웃의 범위를 고려하여 세분하여야 한다. 예를 들어, 다음 식 (12)과 같은 quantization thresholds들을 사용한다고 하자. 이렇게 분할하면, 3절에서 ASE에 의한 제어계의 decoder에 대해서, 시스템은 불연속화된 입력공간에서 13,122(=9×9×18×9) 가지의 상태를 가지게 된다. 이것은 13,122개의 채널을 가지는 거대한 정보 통로를 나타내게 되고, 정보 전달에 과도한 비용을 부담하는 결과가 된다.

1) x : ±0.27, ±0.8, ±1.34, ±1.87, ±2.4 [m]
2) $\dot{x}$: ±0.1, ±0.5, ±10.0, ±1000.0, ±∞ [m/sec]
3) θ : 0, ±1.3, ±2.7, ±4.0, ±5.3, ±6.7, ±8.0, ±9.3, ±10.7, ±12.0 [deg]
4) $\dot{\theta}$: ±10.0, ±50.0, ±400.0, ±2400.0, ±∞[deg/sec] (12)

위에서 고려한 입력변수를 불연속화하는 단위 구간의 크기를 줄임으로서 개별 입력변수의 분할 구간의 수가 늘어나고, 그런 개별 변수들에 대한 구간들을 decoder 방식에 의한 1차원적으로 조합하여 시스템의 상태를 decoding함으로써 과도한 규모의 정보 통로가 생기게 된 것이다. 여기서, 연속적인 입력변수의 불연속화와 신경망의 학습일반화 개념에 관한 CMAC의 변환 단계들을 고려하면 지금까지 살펴 본 사항들을 반영할 수 있는 방안을 찾을 수 있다.

그림 2의 제어계의 decoder가 하는 역할은 CMAC의 S-to-M 변환으로 볼 수 있다. N개의 연속적인 입력변수로 구성된 입력벡터, S는 CMAC에서 각 입력변수에 해당하는 N개의 입력변수축, s_i들로 구성되는 N차원 입력공간(receptive fields)[20]을 정의한다. $S = (s_1, s_2, \cdots, s_N)$. 각 입력변수축은 CMAC을 설계할 때 적당한 해상도(resolution)에 따라 불연속화되기 때문에, 입력변수축은 입력변수값 범위, 즉 입력변수축의 크기를 해상도로 나눈 수만큼의 마디로 분할된다. 그에 따라 입력공간은 유한한 개수의 영역들로 구성된다.

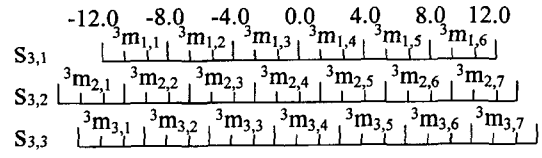

그림 4. CMAC의 S-to-M 변환에 의한 입력변수 s_3의 세 계층으로의 변환

CMAC에서는 각 입력변수가 K개의 중간변수로 변환되므로, 각 입력변수축은 K개의 중간변수축(quantizing function)으로 변환된다. 각 중간변수축도 불연속화되는데, 중간변수축의 해상도(quantizing interval)는 입력변수축의 해상도의 K배로 정의된다. 같은 입력변수축에 대한 K개의 중간변수축들은 서로 입력변수축의 해상도만큼 전위(offset)되어 있다. 그래서, 해상도에 따라, 같은 입력변수축에 대한 K개의 중간변수축들이 분할된 마디의 개수는 보통 서로 다르다.

그림 4는 식 (3)에서 세 번째 입력변수인 θ를 신경망의 계층(layer)의 개념에 따라 세 개의 중간변수로 변환하고 각 중간변수들을 다시 불연속화한 것을 보여준다. 집합($^3m_{1,1}$, $^3m_{1,2}$, $^3m_{1,3}$, $^3m_{1,4}$, $^3m_{1,5}$, $^3m_{1,6}$)는 제 1계층 중간변수 $s_{3,1}$를 나타내고, 집합($^3m_{2,1}$, $^3m_{2,2}$, $^3m_{2,3}$, $^3m_{2,4}$, $^3m_{2,5}$, $^3m_{2,6}$)은 두 번째 계층 중간변수 $s_{3,2}$, 집합($^3m_{3,1}$, $^3m_{3,2}$, $^3m_{3,3}$, $^3m_{3,4}$, $^3m_{3,5}$, $^3m_{3,6}$, $^3m_{3,7}$)은 세 번째 계층 중간변수 $s_{3,3}$를 나타낸다. 특기할 것은 두 번째와 세 번째 계층에서 불연속화의 구간의 개수가 늘어난 것이다. 즉 $s_{3,1}$은 성분이 여섯 개인데 반해, 나머지 중간변수들은 성분의 개수가 일곱 개이다. 이것은 CMAC의 변환의 특성에 의한 것이다.

위 그림에서, 예를 들어, 성분 $^3m_{1,1}$은 입력변수 s_3에 대한 첫 번째 계층의 중간변수 $s_{3,1}$에서 변수 값의 범위[-12.0, -8.0)에 해당하는 이름표이다. 이 이름표에 의해 표시되는 변수 값의 범위는 위의 quantize thresholds에 따라서 다음과 같은 세 개의 소구간으로 세분화되어 있다: [-12.0, -10.7), [-10.7, -9.3), [-9.3, -8.0). $s_{3,2}$에 대해서, $^3m_{3,2}$는 범위 [-10.7, -6.7)에 대한 이름표이고, $s_{3,3}$에 대해서, $^3m_{3,2}$는 [-9.3, -5.3)에 대한

이름표이다. ${}^3m_{2,1}$은 그에 대한 왼쪽 경계는 정해져 있지 않지만, 실제 입력변수의 값이 -12.0 이하에 대해서는 정의되어 있지 않기 때문에 개의하지 않아도 되며, 단지 s_3에 대한 두 번째 계층에서 입력변수의 값 범위[-12.0, -10.7)를 담당할 뿐이다. 마찬가지로, ${}^3m_{3,1}$은 s_3에 대한 세 번째 계층에서 입력변수의 값 범위 [-12.0, -9.3)를 담당한다. 그래서, 연속적인 입력공간에서, 예를 들어, s_3 = -8.4이면, 불연속 입력공간에서는 다음과 같은 세 개의 중간변수 성분들의 집합으로 입력변수가 표기된다(${}^3m_{1,1}$, ${}^3m_{2,2}$, ${}^3m_{3,2}$).

학습일반화의 근원이 되는 입력 공간에서의 이웃 개념에 대해서 살펴보자. 예를 들어, s_3 = -6.0이면, 이 변수 값과 관련되는 중간변수 성분들의 집합은 (${}^3m_{1,2}$, ${}^3m_{2,3}$, ${}^3m_{3,2}$)이다. 위의 s_3 = -8.4에 대한 중간변수 성분들과 비교하면, ${}^3m_{3,2}$가 공통 성분이다. 이 성분으로 인해, s_3 = -6.0을 하나의 성분으로 가지는 아무 시스템의 상태는 s_3 = -8.4을 성분으로 가지는 아무 시스템의 상태와 입력상태공간에서 서로 이웃 상태이다. 이 두 성분으로 인해, 두 시스템의 출력 값에 입력 변수 값들 간의 유사성이 반영되어 나타난다는 의미이다.

하나의 입력변수를 불연속화하는데 각 각 세 개의 계층이 연루되기 때문에, 각 계층 별로, 시스템의 상태는 독립적으로 정의된다. 그래서, 첫 번째 계층에서는 시스템의 상태가 162가지로 정해지고, 두 번째와 세 번째 계층에서는 각각 448가지로 정해진다. 이렇게 되면, 입력 정보 전달에 동원되는 채널 수는 원래의 162개 외에, 448 채널짜리 통로 두 개가 늘어난 결과가 된다.

신경망을 이용하는 제어에서는 학습일반화의 특성을 이용하여 학습의 속도를 빨리 하는 이점을 추구한다고 하더라도, 지나친 채널 수의 증가는 바람직하지 않다. 그래서, 학습의 일반화 특성을 살리면서 정보 통로의 규모가 과도하게 커지지 않도록 CMAC의 S-to-M 변환에서 입력변수를 불연속화하여야 한다.

이런 문제들을 고려하여, [11]에서 제시된 ASE로 구성한 제어계에서 decoder와 ASE의 메모리 요소의 기능을 CMAC으로 대체한 제어계를 구성하려고 한다. 그림 5는 ASE와 CMAC을 통합한 운반차-막대 제어계의 개략을 보여준다. 그림 2의 제어계와 비교하면, decoder의 역할은 CMAC의 S-to-M 변환 단계에 해당한다. ASE에 전달되는 학습 가중치는 CMAC의 M-to-A와 A-to-p 변환 단계가 합쳐진 것으로 간주할 수 있다.

ASE 제어계에서는 decoder에 의해 단 하나의 채널과 연관된 가중치만이 ASE로 전달되지만, 그림 5의 제어계에서는 CMAC에서 사용하는 정보 전달 계층의

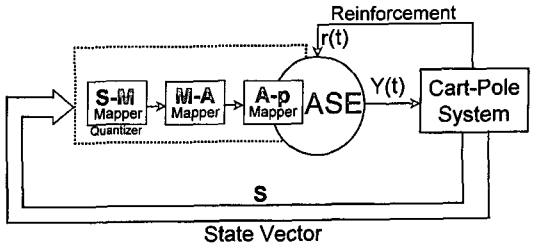

그림 5. 그림 2의 decoder와 ASE의 기능 일부를 CMAC으로 대체하여 구성한 제어계의 개략도

수만큼의 채널로 상태 정보가 전달되므로, K개의 가중치가 ASE로 전달된다.

그림 5의 제어계에 의하면, 시스템에 전달되는 출력 $y(t)$는 다음과 같이 된다:

$$y(t) = w_{1I}(t)a_{1I}(t) + w_{2I}(t)a_{2I}(t) + ,\cdots, + w_{KI}(t)a_{KI}(t) + \text{noise}(t) \tag{13}$$

여기서, $a_{KI}(t)$은 K번째 계층에 의해 구성된 통로 중에서 I번째 채널을 의미하고, $w_{KI}(t)$는 시각 t에 $a_{KI}(t)$에 연결된 가중치이다. 어떤 수학 모델을 사용한 확률 시험 신호 noise(t)를 사용하는가에 따라서 학습이 영향을 받았기 때문에, 컴퓨터 시뮬레이션에서 확률 시험 신호 없이 행하였다.

5. 결과 및 토의

[11]에서 사용된 시스템에 대한 운동방정식, 식 (1) 및 식 (2)를 사용하였고, 시스템의 초기 상태는 [7]에서 제안한 그림 1에서 보는 바와 같은데; 막대가 세워져 있는 운반차는 트랙의 중앙으로부터 x = 1.0 m에서 $\dot{x}$ = 1.0 m/sec로 움직이고 있으며, 막대는 수직으로부터 5.73도(0.1 rad)기울어진 상태에서 11.46도/sec(0.2 rad/sec)의 속도로 기울어지고 있다: 그래서 학습대상 상태는 다음과 같다: S = (1.0 m, 1.0 m/sec, 5.73 deg, 11.46 deg/sec). 강화 학습에 필요한 시스템 매개변수들은 다음과 같다: 식 (10)의 α = 1000, β = 0.5, γ = 0.95, 식 (11)의 δ = 0.9[11]. 시스템의 상태를 구하기 위해 Runge-Kutta 4th Order 알고리즘을 이용하여 미분방정식을 수치적으로 풀었는데, Δt는 0.02초로 하였으며, 한 학습 기간은 100,000 시각 단계로 정하였다.

그림 6은 학습 대상 시스템 상태에 대해 ASE로 구성된 그림 2 제어계가 학습한 성능을 보여준다. 이 그림에 의하면, 746회의 학습을 시도한 다음에 ASE의 메모리에 제대로 된 가중치들이 저장되어, 100,000회 이상의 시각 단계 동안에도 실패없이 유지되어, 제

어계가 주어진 시스템의 상태를 제대로 학습했음을 알 수 있다.

그림 7에서는, 학습 대상 시스템 상태에 대해 그림 5의 CMAC 제어계에 의한 학습 성능을 그림 2의 ASE 제어계의 학습 성능을 보여주는 그림 6과 비교한다. CMAC은 식 (12)의 quantization thresholds와 $K=3$으로 구성하였다. CMAC은 25번의 학습만에 주어진 상태를 성공적으로 학습하여서, 그림에서 CMAC으로 표시된 것과 ASE로 표시된 것을 비교해보면,

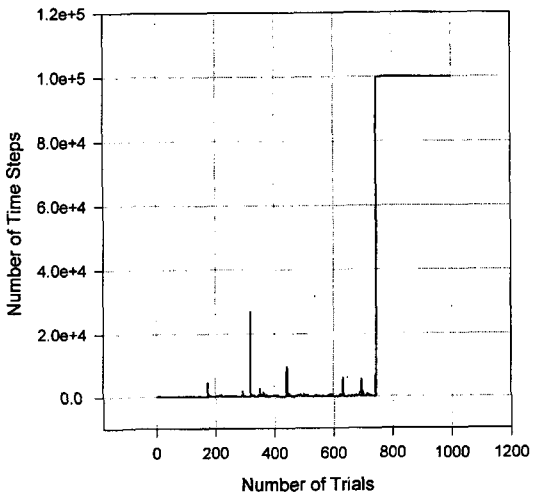

그림 6. ASE 제어계가 운반차-막대 시스템의 어떤 상태에 대해 학습을 시도한 회수(trials)와 각 학습 시도가 실패로 끝날 때까지의 시각 단계(time-steps)

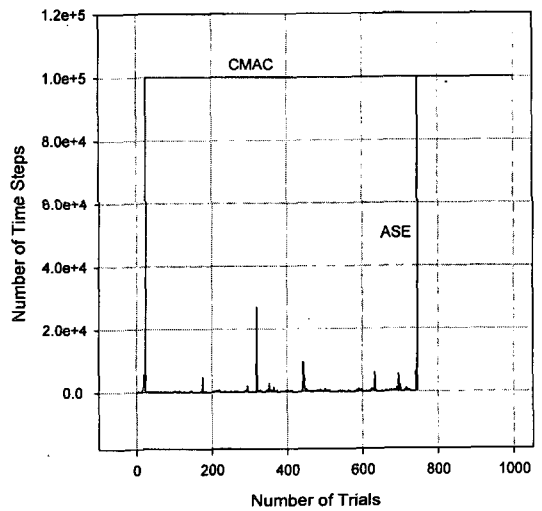

그림 7. K=3을 갖는 CMAC 제어계가 운반차-막대 시스템의 어떤 상태에 대해 학습을 시도한 회수(trials)와 각 학습 시도가 실패로 끝날 때까지의 시각 단계(time-steps)를 그림 6의 ASE 제어계와 비교한다.

CMAC으로 표시된 것이 훨씬 더 적은 수의 학습 시도만에 성공한 것을 알 수 있다.

학습 속도에 현격한 차이가 있는 것에 대해서 검토해 보자. 우선 CMAC의 경우에 입력 변수를 불연속화하는 마디들의 크기가 ASE의 경우에 비해 훨씬 작다. 이것은 식 (3)과 식 (12)에 의한 불연속화 마디의 크기를 비교해 보면 쉽게 알 수 있다. 그래서 입력 변수 값이 ASE의 decoder에 의해서 보다는 CMAC의 S-to-M 변환에 의해서 훨씬 더 정확하게 나타내졌고, 그로 인해 학습 속도가 빨라졌을 것으로 이해할 수 있다.

다음으로, CMAC에 동원된 메모리 개수와 ASE 가중치의 개수는 큰 차이를 보인다. 소요 메모리 량을 비교하면, ASE의 경우에 162 채널에 대해서 162개인데 비해, CMAC은, 세 계층에 대해서, 162, 488, 및 488 개의 채널에 대해서 총 1,138개이다. 더 많은 메모리를 학습에 동원함으로써, 더 빨리 학습할 수 있었을 것이다.

학습에 더 많은 메모리를 소요하여 학습속도가 빨라지긴 했지만, ASE 제어계와 비교하면 메모리 소요량의 측면에서는 비용을 더 치르는 결과인 셈이다. 그래서, 학습 속도를 빨리 하되 메모리 소요의 부담을 줄이기 위해서, 다른 매개변수로 정의되는 CMAC으로 제어계를 구성하였다.

CMAC의 메모리 소요량을 줄이기 위해서, 다음에서 보는 식 (14)과 같은 quantization thresholds를 갖

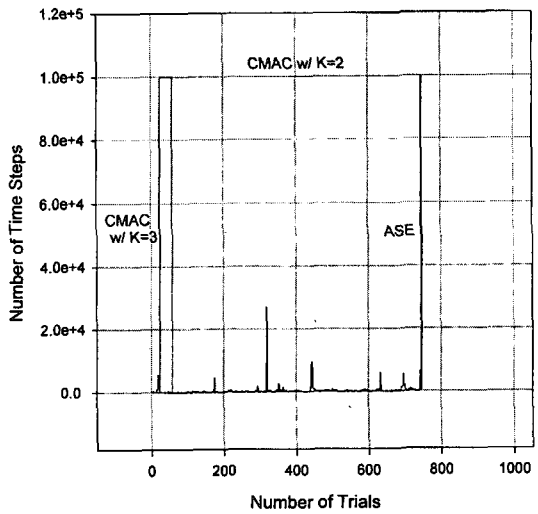

그림 8. K=2를 갖는 CMAC 제어계가 운반차-막대 시스템의 어떤 상태에 대해 학습을 시도한 회수(trials)와 각 학습 시도가 실패로 끝날 때까지의 시각 단계(time-steps)를 그림 6의 ASE 제어계 및 그림 7의 K=2를 갖는 CMAC 제어계와 비교한다.

고 $K=2$를 갖는 CMAC으로 제어계를 구성하였다. 그림 8에서는, 학습 대상 시스템 상태에 대해, ASE 제어계에 의한 학습 성능, $K=3$을 갖는 CMAC 제어계의 학습 성능 및 $K=2$를 갖는 CMAC 제어계의 학습 성능을 비교한다.

1) x : ± 0.8, ± 2.4 [m]
2) $\dot{x}$: 0, $\pm\infty$ [m/sec]
3) θ : ± 5.0, $\pm\infty$ [deg]
4) $\dot{\theta}$: 0, $\pm\infty$ [deg/sec] (14)

그림에서 보듯이 $K=2$를 갖는 CMAC 제어계가 ASE 제어계보다 훨씬 빠르기는 해도 $K=3$을 갖는 CMAC 제어계의 학습 속도에는 못 미친다. 그러나, 소요 메모리를 비교해보면, $K=3$을 갖는 CMAC은 1,138개의 메모리를 소요한데 반해, $K=2$를 갖는 CMAC의 메모리 소요량은 첫 번째 계층에 대해서 36(=3×2×3×2)개와 두 번째 계층에 대해서 144(=4×3×4×3)개, 합해서 180개이다. 그래도, ASE에 의한 메모리 162개에 비해서는 약간 많다.

K=3을 갖는 CMAC 제어계의 소요 메모리 량보다 $K=2$를 갖는 CMAC 제어계의 소요 메모리 량이 대폭 줄어든 것은, 우선 입력 변수들에 대한 quantization thresholds들이 다르기 때문이다[22]. 즉, K=3를 갖는 CMAC에 대한 quantization thresholds 식 (12)에 비해서, 식 (14)에 의한 quantization은 입력 변수 한 마디가 훨씬 더 넓은 범위의 입력 변수 값들을 담당하게 됨으로써, 입력 상태공간을 나타내는 영역의 수가 대폭 줄어들게 된 것이다.

그림 6에서 ASE 제어계는 746번의 학습을 했고, 그림 7에서 $K=3$를 갖는 CMAC 제어계는 25번의 학습을 한데 반해, 그림 8에서 $K=2$를 갖는 CMAC 제어계는 59번의 학습을 거쳤다. 앞에서 ASE 제어계의 가중치 개수와 $K=3$를 갖는 CMAC 제어계의 소요 메모리 개수를 비교하여 학습에 동원되는 메모리 개수의 대소에 따라 학습속도가 영향을 받을 것으로 추정하였다. 그러나 단순히 소요 메모리의 대소만이 학습속도에 영향을 미치는 것이 아님을 ASE 제어계의 소요 메모리 수(162개)와 $K=3$를 갖는 CMAC 제어계의 메모리 개수(180개)를 비교하면 알 수 있다. 소요 메모리 량이 비슷하다는 것은 또한, 식 (3)과 식 (14)을 비교하면 알 수 있듯이, 입력공간을 불연속화하는 quantization thresholds도 비슷하다는 것을 의미한다.

이제, 비슷한 소요 메모리 량에 대해서 ASE 제어계와 CMAC 제어계가 현격한 학습속도 차이를 보이는 이유를 검토해 보자. 그러기 위해서 우선, 적응비평학습 방법에 의해 메모리의 내용에 어떤 변화가 있었는지를 살펴보자. ASE 가중치 162개중에서 154개의 메모리 값들이 초기값과는 다른 값을 보유하고 있었다. 이것은 긴 학습 기간에 거의 대부분의 가중치들이 변한 것으로 생각할 수 있다. 그러나, $K=3$을 갖는 CMAC의 메모리 내용은 1계층에 의한 162개중에서 76개, 2계층 488개중에서 50개, 3계층 488개중에서 61개만이 그 내용이 변하였고 나머지 메모리의 내용은 초기값이 그대로 남아 있었다. $K=2$을 갖는 CMAC의 메모리 내용은 1계층에서는 25개, 2계층에서는 44개의 메모리 내용이 변했다. 또한, 두 경우에, 값이 변한 메모리들은 무리를 지어 있다. 이것은 25회와 59회라는 비교적 짧은 학습 회수로 인해서 메모리 값들이 변화될 기회가 적었고, 메모리 공간의 대부분의 영역에 학습의 영향이 미치지 않은 것으로 판단된다.

CMAC 메모리 중에서 값이 변한 메모리의 량이 전체 메모리 량에 비해서 적고, 변한 메모리들이 무리를 지어 있다는 사실은, 학습 대상 상태에 대해서는 학습이 빨랐지만, 제어계의 입력 공간 전체에 대한 학습 효과는 부진할 것으로 예상할 수 있다. 이런 예상을 확인하기 위하여, 정해진 학습 상태에 대한 훈련을 마친 제어계를 다음과 같은 두 개의 다른 상태에 대해서 시험해 보았다: 즉, S_{T1} = (2.0 m, 0.4 m/sec, 11.46 deg, 22.92 deg/sec)와 S_{T2} = (-2.0 m, -0.9 m/sec, 11.46 deg, -57.3 deg/sec). ASE 제어계는 S_{T1}에 대해서는 실패 없이 곧바로 성공했고, S_{T2}에 대해서는, 1,700번의 학습을 시도한 뒤에 성공하였다. 그러나, K=2나 K=3를 갖는 CMAC 제어계는 두 상태에 대해서 2,000회 이상의 학습으로도 성공하지 못했다. 그래서, CMAC의 빠른 학습효과를 더 넓은 입력공간에 파급하기 위해서, 중간변수의 개수를 대폭 늘여서 $K=12$를 갖는 CMAC을 구성하고 제어계를 시험해 보았으나, 학습속도만 약간 빨라질 뿐, 위에서 예를 든 상태에 대해서는 학습 효과를 보이지 못했다.

그러므로, 주어진 학습 대상을 빨리 학습하는 것은 소요 메모리 량이나 입력변수에 대한 quantization threshold에 의한 영향보다는, 무엇보다도, CMAC의 학습일반화 특성에 의한 영향이 지배적이었을 것으로 생각한다. 이렇게 생각하는 이유는, 위에서 살펴본 대로, CMAC의 메모리 중에서 학습에 의해 메모리 내용이 바뀐 것들이 전체 메모리 규모에 비해 작고, 그것들도 무리를 지어 분포하고 있으며, 초기 조건이 다른 운반차-막대 시스템에 대해서는 학습 효과를 발휘하지 못하는 사실 등에서 알 수 있다.

학습속도는 빠른데도 불구하고 학습의 효과가 전체

표 1. 그림 2, 그림 7 및 그림 8 제어계들이 다른 초기조건을 갖는 두 학습 대상 상태에 대한 학습 시도 회수

제어계	학습 회수	
	중립	불안정
ASE	31	742
K=3를 갖는 CMAC	2	25
K=2를 갖는 CMAC	41	59
K=12를 갖는 CMAC	10	20

입력공간에 미치지 못하는 것은 [11]에서 제시한 적응비평학습 방법이 CMAC의 학습일반화 특성과 어울리지 않는다고 볼 수 있을 것이다. 학습 효과가 반영된 메모리들이 메모리 공간에서 무리를 이루고 있다는 것은, 또한 이웃[21]으로 볼 수 있는 유사한 입력공간에 대해서 유사한 학습을 반복한 것으로 볼 수 있다. 그래서, 적응비평학습에 의하면, CMAC의 학습일반화의 역효과인 학습간섭[21]의 영향이 학습효과에 더 많이 반영되는 것으로 생각된다.

표 1은 ASE 제어계와 CMAC 제어계가 막대-운반차 시스템의 초기조건이 다른 두 상태에 대해서 성공적으로 학습을 완료할 때까지의 걸리는 학습 시도 회수를 보여준다. 이 표에서 중립 조건은 운반차가 트랙의 가운데에 움직이지 않고 서 있으며 막대도 움직이지 않고 수직하게 서있는 상태, $\boldsymbol{S}$=(0.0 m, 0.0 m/sec, 0.0 deg, 0.0 deg/sec)이며, 불안정 조건에 대한 상태 벡터는 $\boldsymbol{S}$=(1.0 m, 1.0 m/sec, 5.73 deg, 11.46 deg/sec)이다. 어떤 학습 시도에 대해서 시스템의 시각 단계 수가 100,000 이상이면 성공적인 것으로 하였다. 이 표에서 K=2를 갖는 CMAC 제어계가 ASE 제어계보다 약간 느린 학습속도를 보이는 경우가 있지만, [18]에 의하면, ASE/ACE 제어계[11]보다도 CMAC 제어계가 더 빠른 학습속도를 보였다. 이로써, 적응비평학습 방법에 의하면, CMAC의 학습일반화 특성은 학습속도를 빨리 하는데는 기여를 하지만, 학습 효과를 입력공간에 일반화하는 면에서는 특성을 살리지 못하는 것으로 볼 수 있다.

6. 결 론

이 논문에서는 [11]에서 제시한 적응비평학습 방법을 이용하여 운반차-막대 시스템을 제어하기 위하여 CMAC을 기반으로 하는 적응비평학습 제어계를 구성하고, 적응비평학습 기법을 CMAC에 구현하는데 있어서의 변환 문제를 검토하였다. 또한, CMAC 제어계의 학습 속도가 ASE 제어계에 비해 빠르기는 하지만, 입력 공간의 더 넓은 영역에 대해서는 학습효과를 발휘하지 못하는 이유는, CMAC의 학습일반화가 속도를 빠르게 하는 효과를 보이지만, 적응비평학습 방법으로 인한 유사 영역에서의 반복적인 학습이 학습간섭의 영향을 많이 남기기 때문인 것으로 판단된다.

[11]에서 제시한 적응비평학습에서 메모리에 입력되는 학습의 결과는 제어계가 필요로 하는, 예를 들어, 액츄에이터의 운동을 제어하는 직접적인 정보이기보다는 액츄에이터의 제어에 간접적인 영향을 미치는 정보이다. 그래서, 적응비평학습의 결과를 bang-bang 제어에 반영하는 것보다, 액츄에이터의 운동을 가변적으로 제어할 수 있는 적응비평학습 제어 기법이 CMAC에는 더 어울릴 것으로 생각된다. 그렇다면, 학습 속도가 빠른 것 외에, 학습효과의 일반화도 적극적으로 기대할 수 있고, 더 정교한 제어가 가능할 것으로 사료된다.

후 기

본 연구는 한국과학재단 지정 계명대학교 저공해자동차부품기술개발센터의 지원에 의한 것입니다.

참고문헌

[1] V. V. Tolat and B. Widrow, "An Adaptive "Broom Balancer" with Visual Inputs," *IEEE International Conference on Neural Networks*, Vol. II, pp. II-641-II-647, 1988.

[2] B. Widrow and F. W. Smith, "Pattern-recognizing control systems," *Computer and Information Sciences (COINS) Symposium Proceedings*. pp. 288-317, Washington, D. C.: Sparton, Washington, 1963.

[3] J. F. Schalfer and R. H. Cannon, *Proc. of 3rd Congress of the IFAC*. Paper 6CL., 1966.

[4] W. R. Strugeon and W. V. Loscutoff, *Joint Automatic Control Conference*. 857, 1972.

[5] S. Mori, H. Nishihara and K. Furuta, "Control of Unstable Mechanical System: Control of Pendulum," *International Journal of Control*, Vol. 23, No. 5, pp. 673-692, 1976.

[6] G. A. Medrano-Cerda, "Robust Computer Control of an Inverted Pendulum," *IEEE Control Systems*, Vol. 19, No. 3, pp 58-67, June 1999.

[7] S. Geva and J. Sitte, "A Cartpole Experiment Benchmark for Trainable Controllers," *IEEE Control Systems*, pp. 40-51, October 1993,

[8] B. Horne, M. Jamshidi and N. Vadiee, "Neural Networks in Robotics: A Survey," *Journal of Intelligent and Robotic Systems*, Vol. 3, pp. 51-66, 1990.

[9] B. Widrow, "The Original Adaptive Neural Net Broom-Balancer," *Proceedings IEEE International*

Symposium on Circuits and Systems, pp 351-357, 1987.

[10] D. Michie and R. A. Chambers, "BOXES: An experiment in adaptive control," Machine Intelligence 2. E. Dale and D. Michie, Eds, Edinburgh: Oliver and Boyd. pp. 137-152, 1968.

[11] A. G. Barto, R. S. Sutton, and C. W. Anderson, "Neuronlike Adaptive Elements That Can Solve Difficult Learning Control Problems," *IEEE Transactions on Systems, Man, and Cybernetics*, Vol. SMC-13, No. 5, September/October, pp. 834-846, 1983.

[12] Z. Tang, Y. Kobayashi, O. Ishizuka and K. Tanno, "A Learning Fuzzy Network and Its Applications to Inverted Pendulum System," *IEICE Transactions on Fundamentals of Electronics, Communications & Computer Science*, Vol. E78-A, No. 6, pp. 701-707, June 1995.

[13] M. Y. Shieh, C. W. Huang and T. S. Li, "A GA-based Sugeno-Type Fuzzy Logic Controller for the Cart-Pole System," *Proceedings of the 23rd International Conference on Industrial Electronics, Control, and Instrumentations*, Vol. 3, pp. 1028-1033, 1997.

[14] M. Magana and F. Holzapfel, "Fuzzy-Logic Control of an Inverted Pendulum with Vision Feedback," *IEEE Transactions on Education*, Vol. 41, No. 2, pp. 165-170, May 1998.

[15] C. S. Lin, and H. Kim, "CMAC-Based Adaptive Critic Self-Learning Control," *IEEE Transactions on Neural Networks*, Vol. 2, No. 5, pp. 530-533, 1991.

[16] H, Kim and C. S. Lin, "Self-Learning with Adaptive Critic: for Problems with Multiple Control Inputs," *Proceedings, 1991 Artificial Neural Networks Engineering Conference*, St. Louis, MO, Nov. 10-13, pp. 511-518, 1991.

[17] B. Lin, The CMAC-Based Adaptive Critic Self-Learning Scheme: Controlling the Balance of an Inverted Pendulum, MS thesis, Lamar University-Beaumont, 1992.

[18] 권성규, "Cartpole Balancing을 위한 CMAC 제어계," 1998년도 제어계측 · 자동화 · 로보틱스 연구회 합동 학술 발표회 논문집, pp. 170-173, 서울 KOEX, 1998년 3월 27일-28일.

[19] C. S. Lin and H. Kim, "Selection of Learning Parameters for CMAC-Based Adaptive Critic Learning," *IEEE Transactions on Neural Networks*, Vol. 6, No. 3, pp. 642-647, May 1995.

[20] M. Brown and C. Harris, *NeuroFuzzy Adaptive Modelling and Control, Prentice-Hall*, pp. 218-258, 1994.

[21] J. S. Albus, "Data Storage in the Cerebellar Model Articulation Controller(CMAC)," *Journal of Dynamic Systems, Measurement, and Control, Transactions of the ASME, Series G*, Vol. 97, No. 3, pp. 228-233, September 1975.

[22] 권성규, "고차원 CMAC 문제의 소요 기억량 감축," 퍼지 및 지능 시스템학회 논문지, 제6권, 제3호, pp. 3-13, 1996, 9.

권 성 규 (Sung-Gyu Kwon)

1980년 : 연세대학교 기계공학 학사
1983년 : 연세대학교 기계공학 석사
1990년 : Louisiana State University, 기계공학 박사
1991년 : 한국원자력연구소 원격장치기술실 선임연구원
1995년 : 계명대학교 기계공학과 조교수

관심분야 : CMAC 응용 및 지능제어, 기구설계