강화 학습에 기초한 로봇 축구 에이전트의 설계 및 구현

김 인 철†

요 약

로봇 축구 시뮬레이션 게임은 하나의 동적 다중 에이전트 환경이다. 본 논문에서는 그러한 환경 하에서 각 에이전트의 동적 위치 결정을 위한 새로운 강화학습 방법을 제안한다. 강화학습은 한 에이전트가 환경으로부터 받는 간접적 지연 보상을 기초로 누적 보상값을 최대화할 수 있는 최적의 행동 전략을 학습하는 기계학습 방법이다. 따라서 강화학습은 입력-출력 쌍들이 훈련 예로 직접 제공되지 않는 다는 점에서 교사학습과 크게 다르다. 더욱이 Q-학습과 같은 비-모델 기반의 강화학습 알고리즘들은 주변 환경에 대한 어떤 모델도 학습하거나 미리 정의하는 것을 요구하지 않는다. 그럼에도 불구하고 이 알고리즘들은 에이전트가 모든 상태-행동 쌍들을 충분히 반복 경험할 수 있다면 최적의 행동전략에 수렴할 수 있다. 하지만 단순한 강화학습 방법들의 가장 큰 문제점은 너무 큰 상태 공간 때문에 보다 복잡한 환경들에 그대로 적용하기 어렵다는 것이다. 이런 문제점을 해결하기 위해 본 연구에서는 기존의 모듈화 Q-학습방법(MQL)을 개선한 적응적 중재에 기초한 모듈화 Q-학습 방법(AMMQL)을 제안한다. 종래의 단순한 모듈화 Q-학습 방법에서는 각 학습 모듈들의 결과를 결합하는 방식이 매우 단순하고 고정적이었으나 AMMQL학습 방법에서는 보상에 끼친 각 모듈의 기여도에 따라 모듈들에 서로 다른 가중치를 부여함으로써 보다 유연한 방식으로 각 모듈의 학습결과를 결합한다. 따라서 AMMQL 학습 방법은 큰 상태공간의 문제를 해결할 수 있을 뿐 아니라 동적인 환경변화에 보다 높은 적응성을 제공할 수 있다. 본 논문에서는 로봇 축구 에이전트의 동적 위치 결정을 위한 학습 방법으로 AMMQL 학습 방법을 사용하였고 이를 기초로 Cogitoniks 축구 에이전트 시스템을 구현하였다.

Design and Implementation of Robot Soccer Agent Based on Reinforcement Learning

In-Cheol Kim†

ABSTRACT

The robot soccer simulation game is a dynamic multi-agent environment. In this paper we suggest a new reinforcement learning approach to each agent's dynamic positioning in such dynamic environment. Reinforcement learning is the machine learning in which an agent learns from indirect, delayed reward an optimal policy to choose sequences of actions that produce the greatest cumulative reward. Therefore the reinforcement learning is different from supervised learning in the sense that there is no presentation of input-output pairs as training examples. Furthermore, model-free reinforcement learning algorithms like Q-learning do not require defining or learning any models of the surrounding environment. Nevertheless these algorithms can learn the optimal policy if the agent can visit every state-action pair infinitely. However, the biggest problem of monolithic reinforcement learning is that its straightforward applications do not successfully scale up to more complex environments due to the intractable large space of states. In order to address this problem, we suggest Adaptive Mediation-based Modular Q-Learning (AMMQL) as an improvement of the existing Modular Q-Learning (MQL). While simple modular Q-learning combines the results from each learning module in a fixed way, AMMQL combines them in a more flexible way by assigning different weight to each module according to its contribution to rewards. Therefore in addition to resolving the problem of large state space effectively, AMMQL can show higher adaptability to environmental changes than pure MQL. In this paper we use the AMMQL algorithm as a learning method for dynamic positioning of the robot soccer agent, and implement a robot soccer agent system called Cogitoniks.

키워드 : 다중 에이전트시스템(multi-agent system), 로봇 축구 시뮬레이션 게임(robot soccer simulation game), 강화 학습(reinforcement learning)

† 종신회원 : 경기대학교 정보과학부 전자계산학전공 교수
논문접수 : 2002년 2월 2일, 심사완료 : 2002년 3월 18일

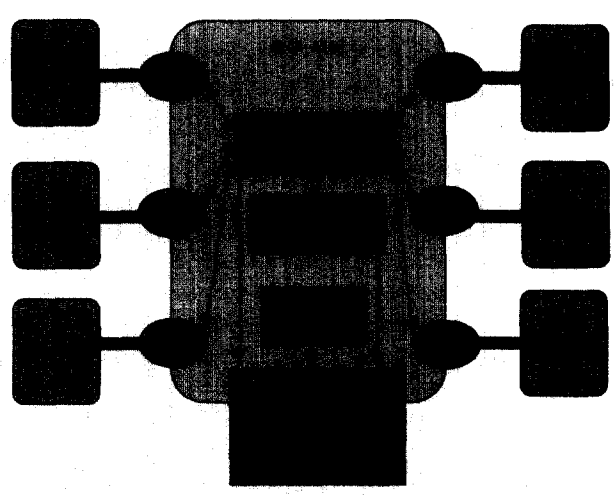
1. 서 론

로봇 축구는 계획, 학습, 조정 등 다중 에이전트를 위한 다양한 이론들이 활발히 연구되고 적용되는 대표적인 다중 에이전트 환경이다. 로봇 축구 환경이 갖는 대표적인 어려운 점들은 여러 선수 에이전트들이 같은 팀끼리는 서로 효과적으로 협조하고 상대팀과는 대적해야 하는 다중 에이전트 환경이고, 하지만 모든 선수 에이전트들이 보고 들을 수 있는 인지범위는 실세계와 마찬가지로 제한적이며 또한 잡음을 내포할 수 있고, 그럼에도 불구하고 각 선수 에이전트들이 끊임없이 실 시간적으로 판단하고 행동해야 한다는 점 등이다. 이와 같은 어려움과 복잡성을 갖고 있는 로봇 축구 환경에서 각 선수 에이전트를 위한 최적 행동 양식을 미리 사람에 의해 설계 해주기는 매우 어렵다[1]. 본 논문에서는 로봇 축구 시뮬레이션 게임 환경에서 각 선수 에이전트들이 전략적으로 유리한 위치를 스스로 결정해 이동하는 행위를 학습하고자 한다. 이러한 행위에 대한 학습은 일반적으로 실제 경기에 임해서만 배울 수 있으므로, 다수의 훈련 예들을 바탕으로 경기 이전에 오프라인으로 학습하는 결정 트리나 신경 망, 베이지안 학습 법보다는 온라인 학습법의 하나인 강화학습법이 보다 적합하다. 하지만 단순한 강화학습 방법들의 가장 큰 문제점은 큰 상태 공간을 갖는 보다 복잡한 환경들에 그대로 적용하기 어렵다는 것이다. 이런 문제점을 해결하기 위해 본 연구에서는 기존의 모듈화 Q-학습방법(MQL)을 개선한 적응적 중재에 기초한 모듈화 Q-학습 방법(AMMQL)을 제안한다. 종래의 단순한 모듈화 Q-학습 방법에서는 각 학습 모듈들의 결과를 결합하는 방식이 매우 단순하고 고정적이었으나 AMMQL학습 방법에서는 보상에 끼친 각 모듈의 기여도에 따라 모듈들에 서로 다른 가중치를 부여함으로써 보다 유연한 방식으로 각 모듈의 학습결과를 결합한다. 따라서 AMMQL 학습 방법은 큰 상태공간의 문제를 해결할 수 있을 뿐 아니라 동적인 환경변화에 보다 높은 적응성을 제공할 수 있다. 본 논문에서는 로봇 축구 에이전트의 동적 위치 결정을 위한 학습 방법으로 AMMQL 학습 방법을 사용하였고 이를 기초로 Cogitoniks 축구 에이전트 시스템을 구현하였다.

2. 로봇 축구 시뮬레이션

로봇 축구 시뮬레이션 시스템은 크게 서버와 클라이언트로 나누어 볼 수 있다. 서버와 클라이언트는 UDP/IP 프로토콜에 의해서 통신교환이 이루어진다. (그림 1)과 같이 축구 서버는 하나의 온라인 서버 프로그램으로서 경기 중 공과 각 선수의 움직임을 시뮬레이트하고, 각 클라이언트와 통신을 통해 현재 경기 상황과 행동 명령을 교환하며, 정해진 규칙에 의해 경기를 조정해나가는 역할을 한다. 각 축구 클라이언트는 자율적으로 동작하는 하나의 선수 에이전트를 나타낸다. 그리고 하나의 경기는 이러한 축구 서버와 각 클라이언트 에이전트간의 지속적인 메시지 교환을 통해 진행된다. 일정한 간격으로 서버에서 클라이언트로 전달되는 메시지에는 공과 각 선수의 위치 뿐 만 아니라 경기장에 대한 모든 정보가 포함되며, 반면에 각 클라이언트에서 서버로 전달되는 메시지에는 해당 선수 에이전트의 동작을 지시하는 명령이 포함된다. 따라서 각 선수 에이전트는 축구서버를 통해 자신의 인식 정보를 받아들이고, 자신의 행동을 수행하게 된다. 그리고 이러한 일련의 과정을 통해 진행되는 축구 경기 상황을 사용자에게 시각적으로 보여주기 위해 2D혹은 3D 그래픽 축구 모니터가 있다[1].

이와 같은 시뮬레이션 시스템이 제공하는 가상의 로봇 축구 환경은 여러 선수 에이전트들이 서로 협조하고 대적해야 하는 다중 에이전트 환경이고, 모든 선수 에이전트들의 인지범위가 실세계와 마찬가지로 제한적이며 또한 잡음을 내포할 수 있고, 그럼에도 불구하고 각 선수 에이전트들이 끊임없이 실 시간적으로 판단하고 행동해야 하는 등 많은 어려운 점들을 포함하고 있다. 따라서 이와 같은 어려움과 복잡성으로 인해 각 선수 에이전트를 위한 최적 행동 양식을 미리 사람에 의해 설계 해주기는 매우 어렵다. 이러한 이유로 그 동안 로봇 축구 시뮬레이션 게임은 많은 에이전트 연구자들에 의해 강화학습을 비롯한 다양한 다중 에이전트 학습 이론들이 연구되고 적용되는 환경으로 이용되어 왔다.



(그림 1) 축구 서버 시스템

강화학습을 로봇 축구 선수 에이전트에 성공적으로 적용한 대표적인 시스템으로 미국 CMU대학의 CMUnit가 있다. CMUnited 축구 에이전트에서는 공 가로채기(ball interception)와 같은 기본적인 행위를 학습하는 하위 계층에서부터 팀 동료 중 누구에게 패스해야 할지를 학습하는 최상위 계

층까지 총 3개의 계층으로 구성된 계층적 학습(layered learning) 구조를 적용하였으며, 특히 이 중에서 패스선택을 위한 온라인 학습(online learning)에 강화학습을 적용하였다[3]. Andou의 Andhill 축구 에이전트에서는 자신이 공을 가지고 있지 않은 경우에 정해진 원래 위치에 단순히 머물러 있지 않고 동적으로 적절한 위치를 찾아 이동하는데 강화학습을 적용하였다. Andhill에서는 Kimura가 제안한 POMDP에 적용 가능한 Q-학습 형태인 SGA 알고리즘을 로봇축구 에이전트에 적용하였다. 이 알고리즘에서는 Q-학습과 더불어 대표적인 귀납적 학습(inductive learning) 방법인 신경망(neural network)을 함께 이용함으로써 큰 상태 공간 문제에 도움을 주었다[6]. KAIST의 로봇 축구 팀 NaroSot에서는 효과적인 지역방어전략을 수행하기 위해 팀 동료들과 협조하는 방법을 모듈화 Q-학습법을 적용하여 학습하였다[8].

3. 강화 학습

강화 학습이란 에이전트가 환경으로부터 받는 누적 보상값을 최대화 할 수 있는 최적의 행동 전략을 학습하는 것이다. 에이전트는 환경 안에서 행동을 실행하고 그 결과에 따라 환경은 보상이나 질책을 제공한다. 강화학습은 목표지향적 학습 방법으로 시행착오, 지연된 보상, 그리고 주어진 환경과의 상호작용을 통하여 학습하는 특성을 가지고 있다. 일반적인 강화학습을 정형적으로 정리하면 다음과 같다. 에이전트는 주어진 특정 상태 s_t에서 정해진 행동전략 π에 따라 현재 실행할 행동을 결정하고 실행한다. 그 결과 환경은 실행한 행동에 대한 스칼라 값의 보상 신호 r_t를 에이전트에게 제공하고 새로운 상태 s_{t+1}로 전이된다. 그러면 에이전트는 다시 이 새로운 상태 s_{t+1}에서 동일한 행동전략에 따라 취해야 할 다음 행동을 선택하고 실행하게 되며, 역시 환경은 그 행동에 상응하는 보상값을 r_{t+1}제공하고 또 새로운 상태로 전이되는 과정이 반복된다[9, 11].

$$V^{\pi}(s_t) \equiv r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + \cdots\cdots = \sum_{i=0}^{\infty} \gamma^i r_{t+i} \quad (1)$$

$$\pi^* \equiv \arg\max_{\pi} V^{\pi}(s), (\forall s) \quad (2)$$

식 (1)은 특정상태 s_t에서 이와 같은 과정이 진행되어간다고 보았을 때 행동 전략 π에 따른 기대되는 누적 보상값을 나타내며, 이것으로 상태 s_t에 대한 가치함수(value function)를 정의한다. 이때 γ는 보상값에 대한 감소율(discount rate)을 나타낸다. 식 (2)는 최적의 행동 전략 π^*을 정의하고 있으며, 이것은 가능한 모든 상태에서 가치함수값 즉 누적 보상 값이 최대가 될 수 있는 행동전략을 의미한다.

Q-학습과 같은 비 모델 기반의 강화학습은 사전에 환경에 대한 별다른 모델을 설정하거나 학습할 필요가 없으며 다양한 상태와 행동들을 충분히 자주 경험할 수만 있으면 최적의 행동전략에 도달할 수 있어 다양한 응용분야에 적용되고 있다. 일반적인 Q-학습 법에서 Q-값 갱신은 식 (3)과 같다[12].

$$Q(s, a) \leftarrow Q(s, a) + \alpha[r + \gamma \max_{a'} Q(s', a') - Q(s, a)] \quad (3)$$

실제 응용분야에서 Q-학습과 같은 강화학습이 겪는 최대의 문제는 큰 상태 공간을 갖는 문제의 경우에 학습에 필요한 Q-테이블의 크기가 너무 커진다는 것이다. 따라서 적절한 시간 내에 가능한 모든 상태와 행동들에 대한 최적의 Q-값에 수렴 할 수 없어 실효를 거두기 어렵다는 점이다[16]. 학습의 수렴속도를 높이는 한 가지 방안으로는 실제 경험보다는 환경에 대한 모델을 기초로 좀더 많은 Q 개선 연산을 수행하는 모델기반 강화 학습 법들이 제안되었다. 하지만 가장 효과적인 상태공간 축소방법 중의 하나로는 Whitehead에 의해 처음 제안된 모듈화 Q-학습 법이다 이 학습방법은 원래 문제의 큰 상태공간에 대해 바로 Q-학습을 전개하지 않고 대신 이 상태공간을 몇 개의 작은 모듈들로 나눈 다음, 모듈별로 개별적인 Q-학습을 전개하고 이들의 Q-학습 결과를 결합함으로써 최적의 행동을 결정하는 방식을 취한다[10]. 하지만 이 방법은 설계자에 의해 할당된 고정 모듈들을 이용하며, 이들을 결합하는 방식 또한 최대 질량 전략(GM : Greatest Mass Strategy)이라고 불리는 단순한 고정적인 방식을 사용함으로써 환경이 급격하게 변화하면 효과적으로 적응하기 어렵다는 단점이 있다

$$a^* \leftarrow \arg\max_{a \in A} \sum_{i=1}^{n} Q_i(s, a) \quad (4)$$

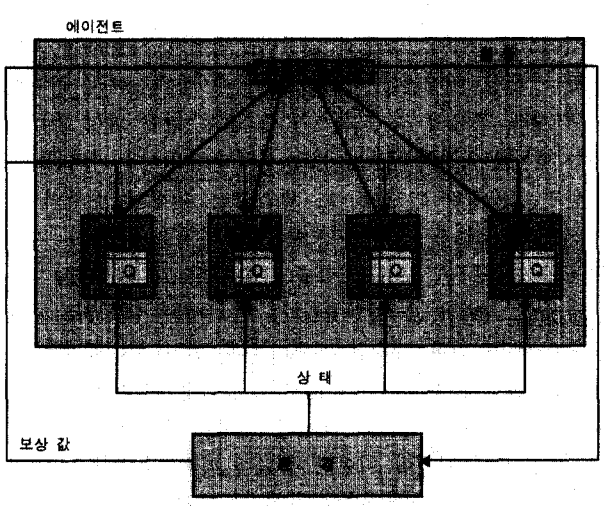
식 (4)는 기존의 모듈화 Q-학습 법에서 사용하는 최대 질량 전략을 나타내는 식으로서, 현재 상태 s에서 가능한 모든 행동 중에서 각 모듈 i의 Q_i 값을 더한 합이 최대가 되는 행동을 최적의 행동 a*으로 선택하는 방식을 나타낸다.

한편 Kohri의 연구에서는 변화하는 환경에 맞추어 에이전트 스스로가 적합한 모듈들의 집합을 만들어 가는 구조인 자동화된 모듈화 학습법(Automatic Modular Q-Learning, AMQL)을 제안했다[2]. 이 방법은 3단계로 자동 모듈 구성을 위한 학습이 진행되는데, 먼저 상태공간을 표현하는 환경 요소들의 부분집합을 임의로 선정하여 각 모듈들을 구성하고, 모듈화 학습을 진행하면서 보상을 받을 수 있는 행동 결정에 기여한 정도에 따라 각 모듈의 적합성을 평가한 뒤, 적합성이 정해진 임계 치 보다 높은 모듈은 잔류하고, 그렇지 못한 모듈들은 새로운 환경 요소들을 선택하여 재구성한다. 이 학습 법은 환경변화에 대응하는 에이전트에게 좀더 높은 적응성을 제공할 수 있으나, 자동

으로 모듈들을 결정하기 위한 별도의 학습비용과 시간을 부가적으로 필요로 하여 로봇축구와 같이 비교적 시간이 짧은 환경에서는 빠른 효과를 기대하기 어렵다.

4. 적응적 중재에 기초한 모듈화 Q-학습

본 논문에서는 순수 Q-학습이 갖는 큰 상태공간 문제를 해결하기 위해 모듈화 Q-학습(MQL)과 그리고 이것을 확장한 자동화된 모듈화 Q-학습(AMQL) 등과 유사한 접근방법을 제시한다[5]. 하지만 이 방법은 AMQL과 같이 에이전트 스스로 내부 모듈들을 학습하도록 하는 방식 대신에 순수 MQL처럼 설계자가 영역지식을 바탕으로 설계해준 고정 모듈들을 사용한다. 하지만 대신 급격한 환경변화에 적응할 수 있도록 모듈들의 결합방식을 동적으로 결정한다. 따라서 적응적 중재에 기초한 모듈화 Q-학습(Adaptive Mediation based Modular Q-Learning, AMMQL)이라 불리는 이 방법은 부가 노력이 많이 필요하고 학습속도가 느린 AMQL보다는 낮은 수준의 적응성을 제공하나, 순수 MQL에 비해서는 높은 적응성을 적은 비용으로 제공할 수 있다는 것이 큰 특징이다. (그림 2)는 n개의 모듈로 구성된 AMMQL의 구조를 보여준다. 에이전트의 행동에 따른 환경으로부터의 보상/강화신호는 개별적인 Q-학습을 전개하는 각 모듈에 전달될 뿐 아니라 각 모듈의 학습결과를 결합하여 최종적으로 실행할 행동을 결정하는 중재모듈에도 전해진다. 그리고 중재모듈은 이와 같은 보상을 이끌어낸 행동을 결정하는데 각 모듈들이 어느 정도 기여하였는지에 따라 행동 결정 함수에 각 모듈의 Q-값을 반영하는 비중을 조정한다. 따라서 AMMQL은 중재모듈에서도 학습이 이루어지는 이중적 학습구조를 취한다.



(그림 2) AMMQL 구조

모듈화하지 않은 상태공간상의 한 상태 $s_k \in S$에 대응되는 각 모듈의 상태를 각각 s_{1k}, s_{2k}, …, s_{nk}이라고 하면 중재모듈의 행동결정함수는 각 모듈의 Q-값을 가중치 에 따라 선형적으로 결합하는 식 (5)과 같다.

$$a^* = \arg\max_{a \in A} (\omega_1 Q_1(s_{1k}, a) + \cdots + \omega_n Q_n(s_{nk}, a)) = \arg\max_{a \in A} \sum_{i=1}^{n} \omega_i Q(s_{ik}, a) \quad (5)$$

그리고 이때 가중치 ω_i들의 합은 1로 한다. 즉, $\omega_1 + \omega_2 + \cdots + \omega_n = 1$.

중재모듈에 의해 선택된 최적 행동 a*을 수행한 결과 환경으로부터 보상 값 R이 주어졌다고 가정하자. 또한 각 모듈 M_i별로 가능한 모든 행동들의 Q-값의 합에 대한 행동 a*의 Q-값의 비율 r_i을 식 (6)과 같이 계산한다고 가정하자.

$$r_i = \frac{Q_i(s_{ik}, a^*)}{\sum_{a_m \in A} Q_i(s_{ik}, a_m)} \quad (6)$$

그러면 각 모듈 M_i의 임시 가중치 ω_i'는 식 (7)와 같이 계산되며, 이때 계수 α는 학습 율을 나타낸다. 그리고 이 임시 가중치들은 식 (8)와 같은 정규화(normalization)를 거쳐 새로운 가중치 ω_i로 갱신된다.

$$\omega_i' \leftarrow \omega_i + \alpha \cdot r_i \cdot R \quad (7)$$

$$\omega_i = \frac{\omega_i'}{\omega_1' + \omega_2' + \cdots \omega_n'} \quad (8)$$

AMMQL은 이와 같은 과정을 반복함으로써 큰 추가비용을 들이지 않고도 환경변화에 더욱 민감한 모듈화 Q-학습을 진행해갈 수 있다

5. 시스템 설계

본 논문에서는 로봇 축구 시뮬레이션 환경에서 자율적으로 동작하는 선수 에이전트인 Cogitoniks 에이전트의 동적 las 위치 결정과 이동 행위를 구현하는데 AMMQL 학습법을 적용하였다. Cogitoniks 에이전트에서는 AMMQL 학습법과 더불어 경기장안의 각 선수들과 공의 위치를 c표시하는데 격자 모델을 이용하여 큰 상태 공간을 줄였다. 즉 Cogitoniks에서는 축구 경기장 필드를 $(6 \times 4) = 24$개의 동일한 크기의 격자 셀들로 나누었다. 또한 현재 공을 가지고 있지 않은 한 선수 에이전트가 이동해야 할 적합한 위치를 결정하기 위해서는 적어도 다음과 같은 요소들을 고려해야 한다.

① 공을 중심으로 한 가장 가까운 팀 동료의 위치
② 공을 중심으로 한 가장 가까운 상대편의 위치

③ 공을 중심으로 한 자신의 위치

이때 각 위치는 위에서 정한 격자모델에 따라 24개의 격자 셀 중에 하나가 된다. 따라서 순수 Q-학습 법을 적용하는 경우, 하나의 상태는 위의 3가지 요소의 조합으로 표현되므로, 가능한 모든 상태들의 집합 즉 상태공간 S의 크기는 $(24 \times 24 \times 24) = 13824$가 된다. 한편, 이 경우 각 선수 에이전트의 선택 가능한 행동이란 바로 경기장내 다른 격자 셀로 이동하는 것을 의미하므로 선택할 수 있는 행동들의 집합 즉 행동공간의 크기는 역시 24이다. 따라서 에이전트가 Q-학습을 위해 가능한 모든 상태-행동 쌍에 대해 Q-값을 저장하는 Q-테이블의 크기는 $(24 \times 24 \times 24) \times 24 = 331776$ 이 된다.

본 논문에서는 이와 같이 큰 전체 상태 공간과 행동공간을 3개의 모듈로 나누었다. 그리고 각 모듈별 상태표현에는 아래와 같이 위에서 언급한 3가지 요소 중에서 단 하나만을 고려하도록 축소하였다.

- 모듈 M_1의 상태 : 공을 중심으로 가장 가까운 팀 동료의 위치
- 모듈 M_2의 상태 : 공을 중심으로 가장 가까운 적의 위치
- 모듈 M_3의 상태 : 공을 중심으로 선수 자신의 위치

이와 같이 나뉘어진 모듈들은 각각 별도의 Q-학습을 전개하고 그 결과를 AMMQL에서 정의한 방식에 의해 결합되어 최적의 행동을 결정한다. 각 모듈의 초기 가중치들은 임의로 배정한다. 이때 각 모듈의 상태 공간 크기는 서로 다른 격자 셀의 개수인 24이며, 또한 행동 공간 크기도 24로 동일 하다. 이렇게 상태 공간을 크게 축소함으로써 각 모듈의 Q-테이블의 크기는 $24 \times 24 = 576$으로 줄어 든다.

환경으로부터 주어지는 보상 값 R은 행동을 실행한 후 일정한 시간이 경과된 후 공의 위치 변동을 고려하여 식 (9-1)과 식 (9-2)와 같이 정의하였다. 식 (9-1)은 일정한 시간이 경과된 후 공이 경기장 밖으로 나갔을 경우의 보상값을 나타내고 식 (9-2)는 공이 경기장안에 있을 경우의 보상값을 나타낸다.

$$R = \frac{R_0}{1 + (\phi - 1) * t_0 / t_{\lim}} \tag{9-1}$$

$$R = \begin{cases} \phi * \dfrac{x_{avg} - x_t}{x_{og} - x_t} (x_{avg} > x_t) \\ -\phi * \dfrac{x_t - x_{avg}}{x_t - x_{lg}} (x_{avg} < x_t) \end{cases} \tag{9-2}$$

공이 경기장을 벗어난 경우를 나타내는 식 (9-1)에서 보상값 R은 볼이 경기장을 벗어난 위치에 따라 달리 결정된다. 즉 상수값 R_0는 볼이 경기장을 벗어난 위치에 따라 상대편 골대에 골인된 경우는 100을, 우리 팀 골대에 골인된 경우는 -100을, 그 밖의 경우는 우리 팀 골 라인에서 상대팀 골 라인쪽으로 진행함에 따라 1에서 25까지 차등적으로 값을 배정해주었다. 이때 ϕ의 값은 10, t_{lim}의 값은 30초로 정해주었다. 한편 공이 경기장안에 있을 경우를 나타내는 식 (9-2)에서는 에이전트가 행동을 취하기 전 공의 처음 위치에서 얼마만큼 전진하거나 후진하였느냐에 따라 보상값이 결정된다. 이때 x_t는 공의 처음 위치를, x_{avg}는 주어진 시간 동안 공의 움직임을 고려한 평균 위치를 나타낸다.

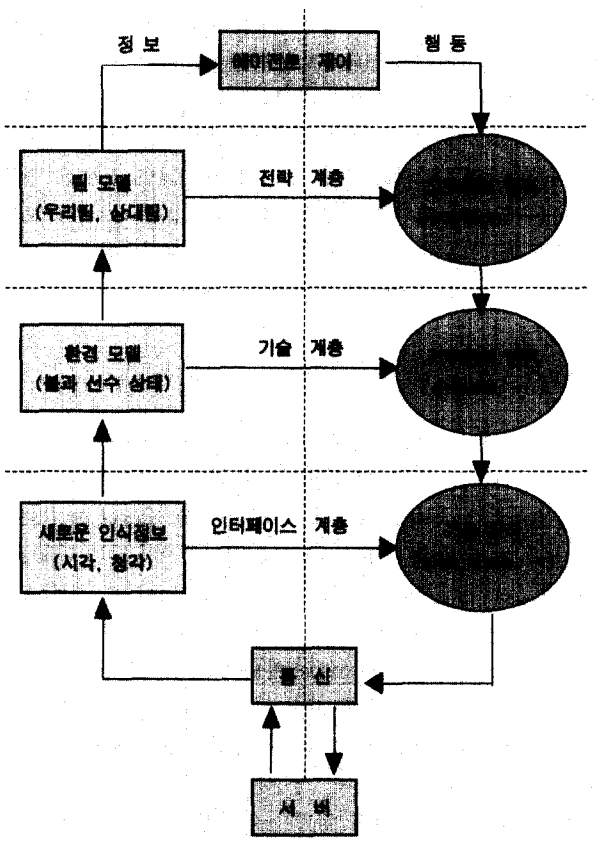
Cogitoniks 축구 에이전트의 위치 이동 행위와 학습은 다음과 같은 과정으로 진행된다 : 매 순간 축구 서버로부터 전달되어 오는 경기의 스냅 샷(snapshot) 정보를 격자 모델로 변환하여 간결한 상태정보를 만들고, 이것을 기초로 각 모듈의 Q-테이블을 참조하게 된다. 중재 모듈이 각 모듈의 가중치를 고려하여 최적의 행동을 결정하면 이를 곧바로 실행하게 되고, 정해진 일정 시간이 경과되면 실행한 행동의 효과에 따라 보상값이 결정된다. 보상값은 다시 각 모듈에 전달되어 Q-값이 갱신됨과 동시에 중재모듈에도 전달되어 각 모듈에 대한 가중치가 새롭게 조정된다.

6. 시스템 구현

본 논문에서 제안한 강화학습법인 AMMQL은 로봇 축구 에이전트인 Cogitoniks의 동적 위치 결정을 위한 행동 제어에 적용되었다. Cogitoniks 에이전트는 공식 RoboCup(Robot WorldCup) Soccer Simulation League에 이용되는 Noda의 Soccer Server 7.0에 기초하여 MS windows환경 하에 MS Visual C++ 6.0로 개발되었다. Cogitoniks는 통신 메시지를 통해 서버로부터 매 150msec 마다 새로운 인식정보를 받아들여 현재 자신이 취할 행동을 결정해 가장 기본 동작들인 Kick(power,angle), Dash(power), Turn(angle), Catch(angle) 중 하나에 해당하는 명령을 서버로 보낸다.

인식정보로부터 최적의 행동을 결정하기 위한 Cogitoniks 에이전트의 내부구조는 (그림 3)과 같이 서로 다른 정보와 역할을 담당하는 3개의 계층으로 구성되어 있다. 맨 아래층은 서버와의 인터페이스(interface)를 담당하는 계층으로서, 서버로부터 자신의 시각과 청각에 인식된 경기장 정보를 받아들이고, 또 역으로 서버에게 4가지 기본 동작 중 하나를 실행하도록 명령을 보내는 역할을 담당한다. 이러한 기능은 축구 시뮬레이션 에이전트에게는 공통적으로 꼭 필요한 필수 기능이다. 인터페이스 계층 위에는 기술 계층(skill layer)이 있으며, 이 층에서는 아래 층에서 제공하는 부분적이고 불완전한 인식정보를 바탕으로 현재 볼과 각 선수들의 위치, 방향, 속도 등 포괄적인 경기상태에 대한 모델을 세우고 이것을 바탕으로 Dribble, Intercept, StopBall, PlaceBall 등의 다양한 기술들 중 어떤 것을 적용할 지 결정한다. 이러한 하나의 기술은 보통 둘 이상의 연속된 기본 동작들로 이루

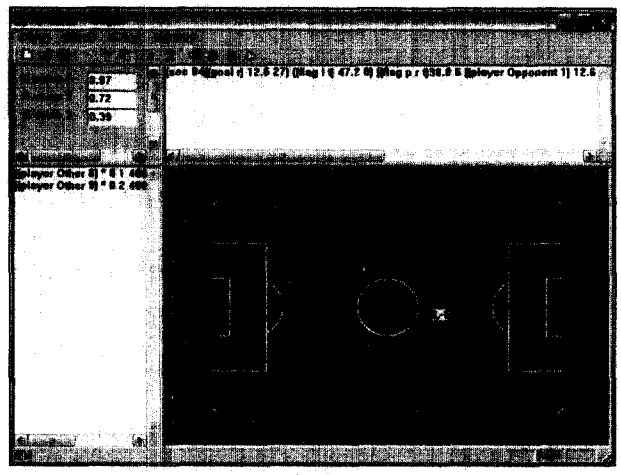
어진다. 맨 위에 위치하는 전략 계층(strategy layer)은 아래 층의 환경모델에서는 볼과 각 선수들의 개별적인 상태 정보를 다루는데 비해 이들을 팀 단위별로 그룹화하여 각 팀의 상태정보를 표현하는 팀 모델을 유지하며 이를 바탕으로 팀 혹은 그룹수준의 전략과 전술을 결정하는 역할을 담당한다. 전략 계층에서 결정되는 고수준의 행위(high-level behavior)들은 각각 단위 모듈들로 구현되어 있으며, 대표적인 모듈들로는 Standard Situation 모듈, GoalKick 모듈, Ball Handling 모듈, Position 모듈 등이 있다.



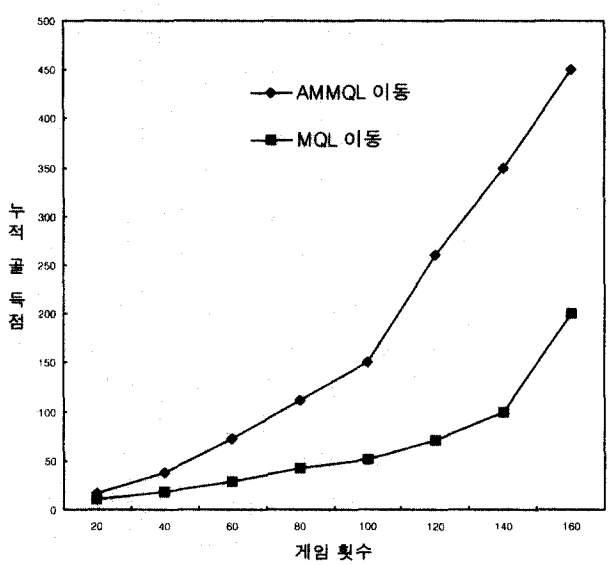
(그림 3) Cogitoniks에이전트의 구조

특히 Position 모듈은 선수 에이전트가 자신이 현재 볼을 가지고 있지 않은 상태일 때 팀 동료들과 상대팀 선수들의 위치 등을 고려해 가장 전략적으로 유리한 곳으로 이동하는 행위를 구현한 것이다. Cogitoniks축구 에이전트의 전략 계층을 구성하는 이 Position 모듈 안에 본 논문에서 제안한 AMMQL강화학습법을 구현하고 적용하였다. (그림 4)는 한 Cogitoniks에이전트의 실행 화면을 보여주고 있다. 왼편 상단은 AMMQL 학습을 위한 4개 모듈의 Q값을 보여주고, 오른편 상단은 축구 서버와의 통신 내용을 보여주며, 오른편 하단은 축구 모니터와 같이 전체 경기 상황을 그래픽으로 보여준다.

한편 본 논문에서는AMMQL 강화학습에 의한 동적 위치



(그림 4) Cogitoniks 에이전트의 실행 화면



(그림 5) AMMQL에 의한 이동과 MQL에 의한 이동 간의 비교

결정이 얼마나 효과적인지를 알아보기 위한 실험들을 전개하였다. 실험을 위해 Cogitoniks 에이전트의 기본 구조 위에 위치결정과 이동을 위한 Position모듈에 AMMQL 강화학습을 적용한 것과 특별한 학습 기능없이 임의로 이동위치를 결정하는 서로 다른 두 가지 전략을 구현하고 비교하였다. 서로 다른 위치 이동 전략의 성능을 직접적으로 평가하기가 쉽지 않아 그 결과로 얻어지는 골 득점으로 대신 평가하기로 하였다. (그림 5)는 서로 다른 이동 전략을 가진 Cogitoniks 에이전트들로 두 팀을 구성한 뒤 이 두 팀 간에 매 5분간 진행되는 게임을 거듭 시행한 결과를 보여주고 있다. (그림 5)의 그래프에 표시된 것은 게임이 거듭 시행되어감에 따라 각기 서로 다른 전략을 가진 두 팀이 얻은 누적 골 득점이다. 게임 횟수가 증가할수록 AMMQL 강화학습에 의해 위치 이동을 하는 에이전트 팀이 MQL에 의해 위치 이동을 하는 에이전트 팀에 비해 골 득점 면에

서 큰 차이로 우수한 성능을 보여주었다. 특히 AMMQL 학습의 경우는 약 90~100회의 게임이 시행된 후, 반면에 MQL학습의 경우는 약 130~140회의 게임이 시행된 후에 각각 골 득점이 현격히 증가하는 현상을 보이는데 이것은 이 시점이 AMMQL 학습을 통해 각 모듈의 평가함수 Q가 최적상태에 수렴하는 순간, 즉 최적 행동 전략에 도달한 순간으로 추정된다. 따라서 보다 더 높은 적응성을 갖는 AMMQL이 MQL에 비해 빨리 최적의 전략에 도달하였음을 보여준 것이다. 또 두 학습법 모두 수렴점을 전후해서 비슷한 성능을 보이리라고 생각했던 예상을 깨고 의외로 MQL 학습에 비해 AMMQL 학습이 전체적으로 높은 성능을 보여주었다.

7. 결 론

본 논문에서는 로봇 축구 시뮬레이션 게임 환경에서 각 선수 에이전트들이 전략적으로 유리한 위치를 스스로 결정하고 이동하는 강화학습 방법을 제안하였다. 일반적인 강화학습 방법들의 가장 큰 문제점은 큰 상태 공간을 갖는 보다 복잡한 문제들에 그대로 적용하기 어렵다는 것이다. 이런 문제점을 해결하기 위해 본 논문에서는 기존의 모듈화 Q-학습방법(MQL)을 개선한 적응적 중재에 기초한 모듈화 Q-학습 방법(AMMQL)을 제안하였다. 종래의 단순한 모듈화 Q-학습 방법에서는 각 학습 모듈들의 결과를 결합하는 방식이 매우 단순하고 고정적이었으나 AMMQL학습 방법에서는 보상에 끼친 각 모듈의 기여도에 따라 모듈들에 서로 다른 가중치를 부여함으로써 보다 유연한 방식으로 각 모듈의 학습결과를 결합한다. 따라서 AMMQL 학습 방법은 큰 상태공간의 문제를 해결할 수 있을 뿐 아니라 동적인 환경변화에 보다 높은 적응성을 제공할 수 있다.

한편 본 논문에서는 계층적 구조의 로봇 축구 에이전트인 Cogitoniks를 설계하고 최상위 계층인 전략 계층의 위치 이동 전략에 앞서 제안된 AMMQL 학습 방법을 적용하였다. 또한 본 논문에서는 MQL학습에 의한 이동 전략을 사용하는 축구 에이전트 팀과의 실험 게임을 통해 동적 위치 이동을 위한 AMMQL학습의 효과를 입증하였다.

참 고 문 헌

[1] Ehsan Foroughi, Fredrik Heintz, Spiros Kapetanakis, Kostas Kostiadis, Johan Kummeneje, Itsuki Noda, Oliver Obst, Pat Riley etc., "RoboCup Soccer Server Manual 7.06," 2001.

[2] Junling Hu, Michael P. Wellman, "Multi-agent Reinforcement Learning : Theoretical Framework and an Algorithm," Proceedings of the Fifteenth International Conference on Machine Learning, pp.242-250, 1998.

[3] Justin A. Boyan, Andrew W. Moore, "Generalization in Reinforcement Learning : Safely Approximating the Value Function, Advances in Neural Information Processing Systems," Vol.6, pp.671-678, 1993.

[4] Kui-Hong Park, Yong-Jae Kim, Jong-Hwan Kim, "Modular Q-Learning based Multi-Agent Cooperation for Robot Soccer," Robotics and Autonomous Systems Vol.35, pp.109-122, 2001.

[5] Kostas Kostiadis, Huosheng Hu, "Reinforcement Learning and Co-operation in a Simulatied Multi-agent System," RoboCup-98 : Robot Soccer World Cup II, pp.366-377, Springer Verlag, Berlin, 1999.

[6] Lesile Pack Kaebling, Michael L. Littman, Andrew W. Moore, "Reinforcement Learning : A Survey," Journal of AI Research Vol.4, pp.147-166, 1996.

[7] Marco Wiering, Rafal Salustowicz, Jurgen Schminhuber, "Reinforcement Learning Soccer Teams with Incomplete World Models," Journal of Autonomous Robots, Vol.1, No.12, 1999.

[8] Micael L. Littman, Anthony R. Cassandra, Lesile Pack Kaelbling, "Efficient Dynamic-Programming Updates in Partially Observable Markov Decision Processes," Brown University Technical Report CS-95-19, 1995.

[9] M. Riedmiller, A. Merke, D. Meier, "Karlsruhe Brainstormers A Reinforcement Learning Approach to Robotic Soccer," Robocup-2000 : Robot World Cup IV, Springer Verlag, Berlin, 2001.

[10] Michael L. Littman, Anthony R. Cassandra, Lesile Pack Kaebling, "Learning Policies for Partially Observable Environments : Scaling up," Proceedings of the Eleventh International Conference on Machine Learning, pp.157-163, San Francisco, CA, 1994.

[11] Noda Itsuki, Matsubara Hitoshi, Hiraki Kazuo, "Learning Cooperative Behavior in Multi-agent EnvironmentA Case Study of Choice of Play-Plans in Soccer," Applied Artificial Intelligence, Vol.12, pp.233-250, 1998.

[12] Norihiko Ono, Kenji Fukumoto, "Multi-agent Reinforcement Learning : A Modular Approach," Proceedings of the Second International Conference on Multi-Agent Systems, AAAI Press. pp.252-258, 1996.

[13] Peter Stone, Richard S. Sutton, Satinder Singh, "Reinforcement Learning for 3 vs. 2 Keepway," RoboCup-2000 : Robot World Cup IV, Springer Verlag, Berlin, 2001.

[14] Peter Stone, "Layered Learning in Multiagent Systems," MIT Press, 2000.

[15] Tomohito Andou, "A Robocup Team which Reinforces Positioning with Observation," Robocup-97 : Robot World

Cup I, pp.373-388, Springer Verlag, Berlin, 1998.
[16] Takayuki Kohri. et al., "An Adaptive Architecture for Modular Q-Learning," Journal of AI Research, 1998.
[17] Veloso M., Pagello E., and Kitano H., "RoboCup-99 : Robot Soccer World Cup III," Springer Verlag, Berlin, 2000.
[18] Weiss G., "Distributed Reinforcement Learning," Journal of Robotics and Autonomous Systems, Vol.15, pp.135-142, 1995.
[19] Weiss G, "Distributed Artificial Intelligence Meets Machine Learning," Springer Verlag, Berlin, 1997.

김 인 철

e-mail : kic@kyonggi.ac.kr

1987년 서울대학교 대학원 전산과학과 (이학석사)

1995년 서울대학교 대학원 전산과학과 (이학박사)

1989년~1995년 경남대학교 전산통계학과 조교수

1996년~현재 경기대학교 정보과학부 전자계산학전공 부교수

관심분야 : 지능형 에이전트, 분산인공지능, 데이터마이닝