논문 2004-41CI-6-8

대전 게임에서 상대방 캐릭터의 행동 패턴을 학습하여 대응하는 신경망 지능 캐릭터

(Neural Networks Intelligent Characters for Learning and Reacting to Action Patterns of Opponent Characters In Fighting Action Games)

조 병 헌*, 정 성 훈**, 성 영 락***, 오 하 령***

(Byeong-heon Cho, Sung-hoon Jung, Yeong-rak Sung, and Ha-ryoung Oh)

요 약

본 논문에서는 일반적인 대전 액션 게임에서 신경망을 이용하여 구현한 지능 캐릭터가 상대방 캐릭터의 행동 패턴을 학습하는 방법을 제안한다. 상대방 캐릭터의 행동 패턴을 학습하기 위하여 상대방 캐릭터의 현재 행동은 물론 과거 행동도 입력받아 학습하게 함으로써, 행동 패턴을 학습하지 않은 경우보다 더 적절한 대응이 가능해진다. 상대방 캐릭터의 행동 패턴의 학습과 더불어 행동의 적절성을 판단하기 어려운 이동에 대한 학습 방법도 제시한다. 제안한 알고리즘의 성능을 평가하기 위하여 실제 게임과 매우 유사한 게임 환경에서 네 가지 행동 패턴을 이용하여 실험하였다. 실험 결과, 모든 행동 패턴에 대하여 지능 캐릭터가 최적 행동을 학습했고, 또한 무작위로 행동하는 캐릭터에 대해서도 이전 방법보다 우수한 성능을 보였다. 이와 같은 결과는 제안한 방법이 상대방 캐릭터의 행동 패턴을 적절히 학습하여 대응할 수 있음을 보여준다. 제안한 알고리즘은 온라인 게임과 같이 캐릭터들이 서로 대결하는 게임들에 쉽게 응용될 수 있다.

Abstract

This paper proposes a method to learn action patterns of opponent characters for intelligent characters. For learning action patterns, intelligent characters learn the past actions as well as the current actions of opponent characters. Therefore, intelligent characters react more properly than ones without the knowledge on action patterns. In addition, this paper proposes a method to learn moving actions whose fitness is hard to evaluate. To evaluate the performance of the proposed algorithm, we experiment with four repeated action patterns in a game similar to real games. The results show that intelligent characters learn the optimal actions for action patterns and react properly against to random action opponent characters. The proposed method can be applied to various games in which characters confront each other, e.g. massively multiple online games.

Keywords: 게임, 인공지능, 신경망, 강화 학습

I. 서 론

최근 들어 컴퓨터의 처리 능력, 특히 그래픽 카드의 처리 능력은 빠른 속도로 발전하고 있다. 이러한 발전 덕분에 컴퓨터 게임에서 CPU가 담당해야 할 그래픽 처리의 부담은 점점 더 감소하는 경향을 보이고 있다. 따라서 CPU는 게임에 필요한 인공지능을 처리할 수 있는 여유를 가지게 되었고, 게임에 다양한 인공지능 기법들이 적용되었다. 기존에 연구된 인공지능 방법으로는 FSM (Finite State Machine), FuSM (Fuzzy State Machine), 신경망, 인공생명, 유전자 알고리즘 등이 있다[1,2,3,4,5,6]. 그러나 전체적인 전략 수립을 목적으로 하는 기존의 방법들은 대전 액션 게임이나 온라인 게임 내의 캐릭터들에게 적용하기 어려운 문제가 있다. 대전 액션

* 학생회원, 국민대학교 전자공학과
(Department of Electronics Engineering, Kookmin University)
** 정회원, 한성대학교 정보공학부
(Division of Information Engineering, Hansung University)
*** 정회원, 국민대학교 전자정보통신공학부
(School of Electrical Engineering, Kookmin University)
접수일자: 2004년8월10일, 수정완료일: 2004년11월15일

게임 등과 같은 장르에서는 게임 전체의 전략을 조율하는 것도 중요하지만, 그 보다는 게임 내의 캐릭터가 주변의 적들의 위치나 적들의 행동에 대하여 적절한 행동을 하도록 지능화 시키는 것이 더욱 중요하다.

지금까지 신경망을 이용하여 대전 액션 게임을 위한 지능 캐릭터를 구현하려는 몇몇 연구가 진행되었다[9][10]. 이들 연구에서는 모두 상대방 캐릭터의 현재 행동만을 고려하여 지능 캐릭터의 행동을 선택한다. 그러나 실제 게임을 하는 사람을 관찰해보면 그 사람만의 독특한 행동 방식이 있게 마련이다. 예를 들어 어떤 사람은 주먹 공격 후에는 반드시 발 공격을 하고, 또 어떤 사람은 특정 거리에서는 필살기만 반복하는 행동 방식을 가질 수 있다. 지금까지의 연구에서는 이와 같은 상대방 캐릭터의 어떤 특정한 반복 행동 패턴에 대응할 수 없는 한계가 있다. 우리는 이와 같은 한계를 극복하기 위하여 신경망 지능 캐릭터가 상대방 캐릭터의 현재 행동뿐만 아니라 과거에 수행한 행동의 순서도 학습하게 함으로써 상대방 캐릭터가 특정 유형의 반복적인 행동 패턴을 보일 때 보다 효과적으로 대응할 수 있는 방법을 제안한다. 또한 공격과 방어 행동과는 달리 현재의 점수에 직접적인 영향이 없어 행동의 적절성을 판단하기 어려운 이동 행동에 대한 학습 방법도 제안한다. 본 논문에서 제안한 방법을 테스트 해보기 위하여 네 가지 행동 패턴을 반복하는 상대방 캐릭터를 이용하여 지능 캐릭터를 학습시켰다. 실험 결과, 모든 행동 패턴에 대하여 지능 캐릭터가 최적 행동을 학습했다. 본 논문에서 제안한 방법이 특정 행동 패턴에 대해서만 우수한지 일반적인 경우에도 우수한지를 살펴보기 위하여 무작위로 행동하는 캐릭터와도 실험을 하였다. 실험결과, 무작위로 행동하는 캐릭터에 대해서도 이전 방법보다 우수한 성능을 보였다. 본 논문에서 제안한 방법은 비록 신경망의 입력 개수를 늘어나게 하여 복잡도를 증가시키고, 학습 시간이 길어지는 단점이 있으나, 상대방 캐릭터의 행동 경향을 파악하여 행동을 할 수 있게 학습으로써 보다 인간과 유사한 지능 캐릭터를 구현할 수 있게 한다.

본 논문의 구성은 다음과 같다. Ⅱ절에서는 게임에 적용된 기존 알고리즘을, Ⅲ절에서는 본 논문에서 제안하는 지능 캐릭터에 대하여 알아본다. 그리고 Ⅳ절에서는 지능 캐릭터를 검증하기 위해 구현한 대전 액션 게임과 실험 결과를 알아보고, 마지막으로 Ⅴ절에서 결론을 맺는다.

Ⅱ. 기존 알고리즘

게임에서 사용되는 인공지능 기법은 FSM (Finite State Machine), FuSM(Fuzzy State Machine)과 같은 전통적인 기법에서부터, 인공생명, 신경망 등에 이르기까지 다양한 기법이 있다[7,8].

현재 가장 널리 사용되는 인공지능 기법인 FSM은 상태들 간의 전이에 의해 통제되는 그래프 내에 유한개의 상태들이 연결되어 있는 규칙 기반 시스템이다[7,8]. FSM은 구현이 쉽고 행동이 정확히 정의되는 장점이 있으나 게임의 진행이 미리 정의된 방식으로만 동작하기 때문에 일정 시간 게임을 한 후에는 쉽게 예측될 수 있어서 게임의 흥미를 반감시키는 요인으로 작용한다. 이러한 단점을 보완하기 위해 FuSM이 사용되고 있다[7]. FSM에 퍼지 이론을 접목한 FuSM의 경우에는 입력과 출력에 퍼지 함수를 적용하여 어느 정도 무작위적으로 동작할 수 있도록 한다. 무작위 요소가 포함되어 있기 때문에 게임에서의 상대방이 동일한 상황에서 다른 행동을 할 가능성이 있어서 상대방의 행동을 예측하기가 어렵게 된다. 그러나, FSM과 FuSM은 캐릭터의 상태의 수가 많아지게 되면, 상태 다이어그램을 정리하기도 어렵고, 프로그램이 급격하게 복잡해지는 단점이 있다. 또한 FSM과 FuSM 모두 새로운 행동 패턴을 추가하기 위해서는 새롭게 프로그램을 해야만 하는 단점이 있다.

다음으로, 인공생명은 생명체가 나타내는 현상을 컴퓨터, 로봇 등과 같은 인공 매체 상에 재현함으로써 생명의 일반적인 특성에 대해 연구하는 학문이다. 게임 개발자들은 인공생명 기법을 게임에 적용하려고 오래 전부터 시도를 해왔다. 그러나 지금까지 연구의 초점은 주로 캐릭터들이 군집을 형성하는 게임에서 전체적인 전략을 결정하는 것에 중점을 둔 연구로서 아직 기초적인 단계에 머물고 있다[6].

마지막으로, 신경망은 인간의 신경 체제를 모사한 것으로서, 수많은 간단한 콤포넌트들이 연결된 구조를 가지고, 입력되는 데이터의 패턴에 기반하여 출력을 산출한다[11][12]. 신경망이 게임에 적용되었을 때의 가장 큰 장점으로는, 신경망은 학습 능력을 가지고 있기 때문에 게임을 진행하면서 계속적으로 지능이 향상될 수 있다는 점이다. 지금까지 신경망을 게임에 적용하려는 연구가 많이 진행되고 있으나, 대부분이 오목, Tic-Tac-Toe 등 보드 게임이 주류를 이룬다[5]. 보드 게임은 보드 상에서 움직이는 말들의 전체적인 상황을 인지하여

개개의 말들의 움직임을 결정하는 특성을 가지고 있다. 반면에 본 논문에서 다루는 대전 액션 게임은 하나의 캐릭터가 외부 환경에 적응하여 자신의 행동을 결정해야 하는 특성을 가지고 있다.

우리는 논문 [9][10]에서 신경망을 이용하여 대전 액션 게임을 위한 지능 캐릭터를 구현하는 방법을 제안하였다. 논문 [9]는 지능 캐릭터를 구현하는데 있어서 신경망의 적용 가능성 여부를 알아보기 위한 기초 연구로서, 이 논문에서 우리는 신경망이 지능 캐릭터를 구현하는 데에 있어서 매우 우수하고 실제 응용 가능성이 높은 방식임을 보였다. 그러나 게임에서의 모든 행동이 하나의 시간 단위에서 종료하는 것을 가정하였기 때문에 현재의 대부분의 게임에는 적용할 수 없는 문제점이 있다. 이 문제점을 보완한 논문 [10]은 행동이 하나의 시간 단위에서 종료하지 않고 여러 개의 단계로 이루어지는 게임 모델 하에서 신경망 지능 캐릭터를 구현하였다. 그러나 위의 두 연구에서는 모두 상대방 캐릭터의 현재 행동만을 관찰하여 지능 캐릭터의 행동을 결정하게 하였다. 이러한 방식은 상대방 캐릭터의 행동이 무작위적인 경우에는 큰 무리가 없으나 대부분의 게임이나 인간처럼 특정 행동이 연속해서 발생하는 경우에는 대처할 수 없는 문제점이 있다. 이러한 문제점은 지능 캐릭터의 보다 지능적이고 보다 인간적인 행동 선택 및 대응이라는 측면에서 바람직하지 않은 것이다. 우리는 이러한 문제를 해결하기 위해 본 논문에서 상대방 캐릭터의 행동 패턴을 인식하고 학습하여 적절히 대응할 수 있는 방법을 제안한다.

Ⅲ. 지능 캐릭터의 구현

3.1 신경망의 구성과 학습 방법

본 논문에서 구현하는 지능 캐릭터의 대상이 되는 게임은 논문 [10]에서와 같이 다음을 가정한다.

가정 1 : 캐릭터의 행동이 1 클럭에 종료하는 것이 아니고, 행동별로 여러 단계가 소요된다.

가정 2 : 캐릭터가 어떤 행동을 수행하고 있을 때, 상대방 캐릭터의 공격을 받으면 그 캐릭터의 행동은 무효화된다.

가정 3 : 캐릭터가 수행 중이던 행동을 임의로 취소하거나 다른 행동으로 바꿀 수 없다. 즉, 어떤 행동을 수행 중이었다면 그 행동을 수행하는데 소요되는 시간이 경과하거나 상대방 캐릭터의 공격을 받아 무효화되지 않으면 다른 행동을 시작할 수 없다.

논문 [10]에서는 위와 같은 가정 하에 신경망의 입력으로 상대방 캐릭터의 행동과 행동의 단계, 두 캐릭터 간의 거리를, 그리고 출력으로 지능 캐릭터의 행동을 설정해 학습하였다. 본 논문에서는 이러한 구조에 상대방 캐릭터의 행동 패턴에 따른 대응이 가능하도록 상대방 캐릭터의 과거 행동을 함께 입력한다. 과거 행동의 입력 개수는 상대방 캐릭터의 행동 패턴의 길이와 관계된 것으로서 몇 개가 좋은지 미리 알 수 없기 때문에 특정 개수로 미리 설정한다. 이와 같이 신경망의 구조를 설정하면 상대방 캐릭터의 행동 패턴별로 보다 더 적절한 지능 캐릭터의 행동 선택이 가능해지게 된다. 신경망의 구조는 [10]의 구조에 과거의 행동값들이 입력되는 부분이 추가되었다. 또한 신경망에 입력되는 값들을 결정하는 방법도 같은데 간단히 설명하면 행동은 표 1과 같이 정수로 정의한다, 또한 거리와 행동의 단계도 각각 이산적인 정수로 입력한다.

출력은 지능 캐릭터가 취할 수 있는 행동 개수만큼의 출력 노드가 있으며, 지능 캐릭터는 그 중에서 가장 큰 값을 출력하는 행동으로 자신의 행동을 결정한다. 그림 1은 이와 같은 내용을 종합하여 신경망의 구성을 나타낸 것이다. 그림 1에서 입력은 상대방 캐릭터와 관련된 정보로서 $P_A(t)$는 상대방 캐릭터의 시간 t 에서의 행동을 나타내며 T는 특정 행동이 진행된 정도, D는 지능 캐릭터와 상대방 캐릭터 사이의 거리를 나타낸다. 신경망의 출력 개수는 지능 캐릭터의 행동 개수와 일치하며 N_i는 정지 N_{up}은 위주먹, N_s는 필살기를 나타낸다.

예를 들어 거리 2에서 상대방 캐릭터의 아래주먹공격이 단계 3을 수행 중이고 과거 행동은 <전진, 위주먹, 앉기>였다면, 그 순간의 신경망의 입력은 $P_A(t)$=7, T=3,

표 1. 상대방 캐릭터의 행동별 정수 매핑 값
Table 1. The integer mapping value according to opponent character's actions.

캐릭터의 행동 (A)	정지 (i)	전진 (f)	후진 (b)	점프 (j)	앉기 (d)	막기 (g)	아래주먹 (dp)	위주먹 (up)	아래발 (dk)	위발 (uk)	필살기 (s)
신경망의 입력 P_A	1	2	3	4	5	6	7	8	9	10	11

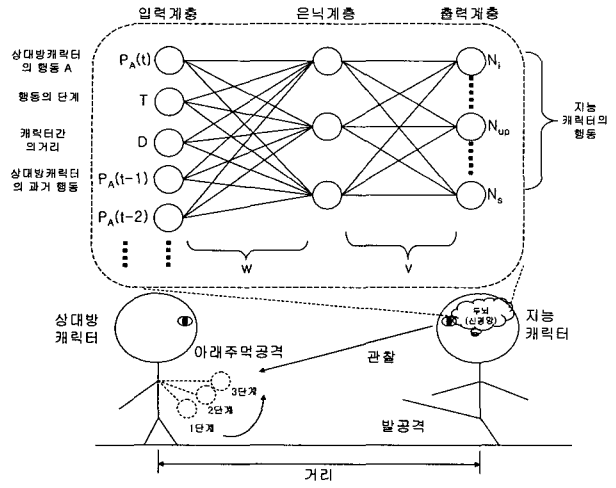


그림 1. 신경망의 구성
Fig. 1. The Structure of the neural network.

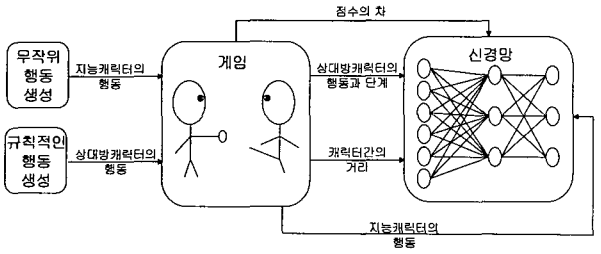


그림 2. 강화 학습 방법
Fig. 2. Reinforcement learning.

D=2, $P_A(t-1)$=5, $P_A(t-2)$=8, $P_A(t-3)$=2이 된다. 결정된 입력값을 신경망에 적용하고 출력값을 계산한다. 신경망은 전방향 신경망 (feedforward neural networks)을 사용하며 일반적인 전방향 신경망과 동일하게 출력을 계산한다[9][10][11].

신경망 링크의 초기 가중치는 무작위적 값으로서, 초기에 신경망의 출력도 무작위 값이 된다. 그러므로 신경망을 이용하여 지능 캐릭터의 행동을 결정하기 전에 먼저 게임 규칙에 대하여 학습을 해야 한다.

학습을 하려면 학습 데이터가 필요하다. 학습 데이터를 얻어서 이를 신경망에 적용시켜 강화 학습하는 방법이 그림 2에 있다. 지능 캐릭터의 훈련 상대인 상대방 캐릭터는 규칙적인 행동을 하도록 하였다*. 학습이 되기 전의 지능 캐릭터의 신경망의 링크 가중치는 무작위 값을 갖고 있다. 즉 지능 캐릭터는 상대방 캐릭터의 행동 양식을 모르는 상태이며 상대방의 행동에 대하여 적절한 행동을 출력하지 못한다. 그러므로 지능 캐릭터의

* 논문 [10]까지는 상대방 캐릭터의 행동도 무작위적이었다.

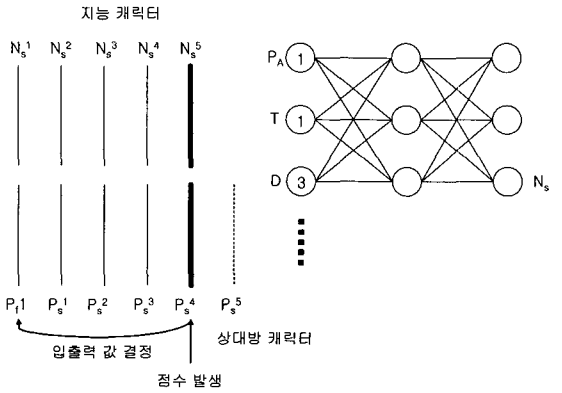


그림 3. 학습할 때의 입출력 값의 결정
Fig. 3. Deciding Input/Output values for learning.

행동은 무작위적으로 선택을 하게 한다. 그림 2에서 보듯이 규칙적으로 선택된 상대방 캐릭터의 행동과 무작위로 선택된 지능 캐릭터의 행동을 게임에 적용한 다음, 게임 상에서 각 캐릭터가 얻은 점수를 강화값으로 이용하여 학습한다. 즉, 주어진 거리에서 상대방 캐릭터의 행동과 그 단계에 대한 지능 캐릭터의 행동이 잘한 것인지 못한 것인지를 지능 캐릭터가 획득한 점수와 상대방 캐릭터가 획득한 점수의 차로써 판단하고, 이를 이용하여 강화학습을 한다. 만약 이 값이 양의 값이면 동일한 상황(상대방 캐릭터의 행동, 상대방 캐릭터와의 거리)에서 해당 행동을 권장하고, 음의 값이면 해당 행동을 선택하지 않게 학습한다. 신경망에 입력되는 행동의 결과 시점을 정하는 것 또한 중요하다. 이는 상대방 캐릭터와 지능 캐릭터의 행동이 결정된 후 실제 게임 규칙에 따라 두 캐릭터의 행동을 진행하고, 두 캐릭터의 행동의 결과로 나타나는 상황을 이용하여 학습을 하기 때문이다. 본 논문에서는 논문 [10]에서 사용한 방법과 동일한 방법을 사용했는데 간단히 설명하면 다음과 같다.

예를 들어 두 캐릭터 간의 거리가 3일 때를 가정한다. 그림 3은 각 캐릭터가 행동하는데 5 클럭이 소요되는 필살기(s)를 시차를 두고 행동한 상황을 나타낸다. 그림에서 N은 지능 캐릭터, P는 상대방 캐릭터를 나타내며, N_S^2는 지능 캐릭터가 필살기(s)라는 행동의 2단계를 수행하고 있는 것을 나타낸다. 상대방 캐릭터가 행동하는 도중에 지능 캐릭터의 행동이 완료되었기 때문에, 만약 지능 캐릭터의 행동이 상대방에게 타격을 주는 유효한 행동이었다면 그 시점에 점수가 발생한다. 그리고 위에서 언급한 가정 2에 따라 상대방 캐릭터의 행동은 중단되어 P_S^5 행동은 무효화된다. 이와 같이 점

수가 발생하면 신경망의 입출력 값이 결정되고, 결정된 값을 이용하여 강화 학습된다. 점수는 지능 캐릭터의 필살기가 끝나는 시점에 발생하지만, 그 점수는 상대방 캐릭터가 전진(f) 행동의 단계 1이 된 시점에서 지능 캐릭터가 필살기 공격을 결정했기 때문에 발생한 것이다. 그렇기 때문에, 이 경우에는 상대방 캐릭터의 현재 행동은 전진 1(표 1 참조), 단계 1, 그리고 두 캐릭터 사이의 거리 3이 입력된다. 또한 상대방 캐릭터가 전진 행동 이전에 행한 일정 개수의 행동들도 입력된다. 출력은 지능 캐릭터의 행동인 필살기로 결정하고, 이 행동의 결과로 얻은 점수의 차(지능 캐릭터의 점수-상대방 캐릭터의 점수)를 이용하여 학습을 하게 된다*.

신경망의 입출력 값이 결정되면 게임 규칙에 따라 계산된 두 캐릭터 간의 점수의 차를 강화값으로 사용하여 신경망을 강화 학습한다. 강화 학습은 오류 역전파 알고리즘을 이용하여 수행된다. 오류 역전파 알고리즘을 이용한 강화 학습 방법을 간단히 설명하면 다음과 같다. 오류 역전파 알고리즘을 이용한 학습은 신경망을 구성하는 각 링크의 가중치를 각 계층별로 수정한다. 먼저 은닉 계층과 출력 계층 사이의 링크의 가중치는 식 (1), (2)에 의해 수정된다[12].

$$\delta_j = z_j \cdot (1 - z_j) \cdot (d_j - z_j) \tag{1}$$

$$v_{ij} \mathrel{+}= \alpha(t) \cdot \delta_j \cdot h_i \tag{2}$$

여기서, d_j는 목표값(desired output)을 의미하며, z_j는 j번째 출력 노드, v_{ij}는 i번째 은닉 노드와 j번째 출력 노드간의 링크 가중치, $\alpha(t)$는 시간에 따른 학습률의 함수로서, 본 논문에서는 학습률 $\cdot e^{-t/\Delta t}$를 사용한다. Δt는 $t_{end}-t_{start}$이고, t_{start}는 학습 시작 시간, t_{end}는 학습 종료 시간이다. 학습률을 이렇게 사용하는 이유는 시간이 지남에 따라 신경망 가중치 조절을 작게 하여 수렴하게 하기 위함이다.

식 (1)에서 d_j는 출력 노드별로 다르게 주어진다. 지능 캐릭터의 행동으로 결정된 출력 노드의 d_j는 점수의 차가 0 이상일 경우에는 점수의 차의 크기에 비례해서 0에서 1사이의 실수로 변환하고, 음수일 경우에는 0으로 정한다. 그 외의 출력 노드들은 학습할 필요가 없으므로 학습을 시키지 않기 위해서 d_j값을 신경망의 출력값 z_j와 같게 한다. 이렇게 하면 δ_j가 0이 되어서 해당 출력의 가중치 변화가 일어나지 않는다. 예를 들어 게임에서 발생할 수 있는 최대 점수차가 5점이고, 그림 3에서 +3점의 점수차가 발생했다면, d_j는 0.6(=3/5)으로 설정된다. 이렇게 함으로써 점수의 차가 양수인 경우 양의 값으로 강화되어 유사한 상황에서 해당 노드는 0.6 근처의 값으로 출력하게 학습된다. 이와 반대로 -3점의 점수차가 발생했다면, 점수차가 음수이므로 목표값을 0으로 설정하여 동일한 상황에서 해당 출력의 행동을 선택하지 않도록 링크의 가중치가 수정된다. 위의 과정을 반복하여 학습이 완료되면, 특정 상황에서 출력 노드들 중에서 가장 큰 값을 출력하는 노드의 행동이 바로 가장 지능 캐릭터가 많은 점수를 획득하는 행동과 일치하게 된다. 그러므로 지능 캐릭터는 신경망 출력에서 가장 큰 출력을 내는 노드의 행동으로 결정한다.

식 (1), (2)를 이용하여 은닉 계층과 출력 계층 사이의 링크의 가중치를 수정한 다음, 입력 계층과 은닉 계층 사이의 링크의 가중치는 식 (3), (4)를 이용하여 수정된다[12].

$$\delta_j = h_j \cdot (1 - h_j) \cdot \left(\sum_{k=0}^{Nh} w_{jk}\delta_k\right) \tag{3}$$

$$w_{ij} \mathrel{+}= \alpha(t) \cdot \delta_j \cdot x_i \tag{4}$$

여기서 x_i는 i번째 입력 노드, h_j는 j번째 은닉 노드, N_h는 은닉 노드의 개수, w_{ij}는 i번째 입력 노드와 j번째 은닉 노드간의 링크 가중치, $\alpha(t)$는 학습률 함수를 나타낸다. 학습이 진행됨에 따라서 지능 캐릭터의 신경망은 특정한 상황 (즉, 상대방 캐릭터의 과거 행동, 현재 행동과 그 단계, 거리)에 따른 자신의 최적 행동이 무엇인지를 학습하게 된다. 본 논문에서 제안한 것과 같이 과거의 행동을 추가로 신경망에 매핑할 경우 지능 캐릭터가 더 좋은 대응 행동을 선택할 수 있는 이유는 다음과 같다.

상대방 캐릭터의 과거 행동이 입력됨으로써 보다 적절한 결정을 하기 위해서는 지능 캐릭터가 획득하는 점수가 상대방 캐릭터의 과거 행동에 따라 변경되어야 한다. 다른 말로 하면 만약에 지능 캐릭터가 획득하는 점수가 오로지 상대방 캐릭터의 현재 행동에만 의존한다면 과거 행동 입력에 따른 장점도 없고, 행동 패턴에 대한 대응도 의미가 없다. 즉, 상대방 캐릭터가 과거에 어떤 행동을 했든 상대방 캐릭터의 현재 행동에 따라 점수가 결정된다면 과거 행동을 입력해서 지능 캐릭터가

* 여기서, 전진(f)은 행동하는데 소요되는 시간을 1 클럭으로 가정하며, 그렇기 때문에 상대방 캐릭터가 전진 행동을 한 후, 바로 다른 행동(필살기)을 수행할 수 있다.

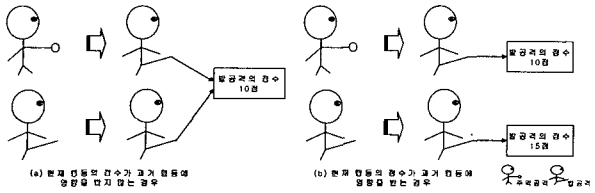


그림 4. 캐릭터의 과거 행동에 따른 현재 행동의 점수
Fig. 4. The scores of current action according to past actions.

더 나은 행동을 결정할 수 없다. 이 내용에 대한 설명을 위해서 그림 4에서 캐릭터의 과거 행동에 따라 현재 행동의 점수 차이가 없는 경우와 있는 경우를 나타냈다.

그림 4(a)는 현재 행동의 점수가 과거 행동에 영향을 받지 않는 경우를 나타낸다. 그림에서 보듯이, 캐릭터가 과거에 주먹공격을 했든, 발공격을 했든 현재 행동이 발공격이면 항상 10점을 획득한다. 이러한 상태에서는 지능 캐릭터가 상대방 캐릭터의 과거 행동을 입력받을 필요도 없고, 또한 입력받는다고 해서 더 나은 행동을 결정할 수 없다. 왜냐하면 점수는 두 캐릭터의 현재 행동에 의해서만 결정되기 때문이다. 반면에 그림 4(b)는 현재 행동의 점수가 과거 행동에 영향을 받는 경우를 나타낸다. 예를 들어, 캐릭터가 주먹공격 후 발공격을 하면 발공격의 점수는 10점이 되고, 발공격 후에 연속해서 발공격을 하면 같은 발공격이지만 이 경우에는 15점을 획득한다고 가정하자. 이와 같은 상태에서는 상대방 캐릭터의 과거 행동이 주먹공격인지, 발공격인지에 따라 지능 캐릭터가 같은 행동을 하더라도 획득하는 점수에 차이가 날 수 있다. 그래서 그림 4(b)와 같이 현재 행동의 점수가 과거 행동에 영향을 받는 경우에는 지능 캐릭터가 상대방 캐릭터의 과거 행동을 입력받아 각각의 경우를 구분함으로써 보다 나은 행동을 학습할 수 있게 된다. 그런데 그림 4(b)와 같이 캐릭터의 과거 행동에 따라 현재 행동의 점수가 차이가 나기 위해서는 그 캐릭터의 과거 행동이 상대하는 캐릭터의 공격 행동에 의해 무효화되지 않아야 한다. 예를 들어, 그림 5와 같이 상대방 캐릭터가 주먹공격의 2단계에서 지능 캐릭터의 발공격을 받아 주먹공격의 3단계가 무효화되면 두 캐릭터는 새로운 행동을 시작하고 그 결과에 따라 점수가 발생한다. 이 때 발생하는 점수는 무효화된 상대방 캐릭터의 과거 행동에는 무관하게 오로지 두 캐릭터의 현재 행동과 단계, 거리에 따라 결정된다.

위에서 설명한 것을 요약하면, 상대방 캐릭터가 과거 행동이 무효화되지 않은 상황에서 어떤 행동을 연속해서 했다면 상대방 캐릭터의 과거 행동을 이용하여 지능

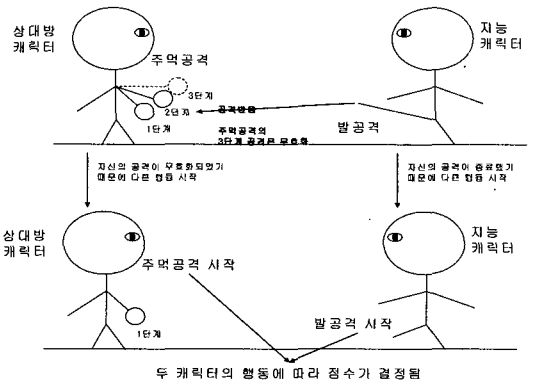


그림 5. 상대방 캐릭터의 행동이 무효화된 후의 점수 결정
Fig. 5. Deciding the score after invalidation of an opponent character's action.

캐릭터가 보다 나은 행동을 결정할 수 있다. 예를 들어 상대방 캐릭터가 주먹 공격의 유효거리 밖에 있다가 전진을 하고, 연속해서 주먹 공격을 한 경우를 가정해보자. 이 경우에는 전진을 하여 주먹 공격의 유효거리 내로 들어가기 때문에 다음에 행하는 주먹 공격으로 점수를 획득할 수 있다*. 그러나 같은 상황에서 상대방 캐릭터가 전진이 아닌 다른 행동을 하고 주먹 공격을 한 경우에는 주먹 공격의 유효거리 밖에 있기 때문에 점수를 획득하지 못한다. 이와 같이 상대방 캐릭터가 과거에 무효화되지 않는 이동 행동을 했다면, 이후에 발생하는 점수가 변경될 수 있다. 이와 같은 이유로 본 논문에서는 두 캐릭터간의 초기 거리를 설정하고 상대방 캐릭터가 이동과 공격 행동을 반복한다고 가정하고 실험했다.

3.2 이동 행동에 대한 학습

3.1절에서 설명한 것처럼 지능캐릭터는 학습 기간 동안에 시도한 여러 가지 행동 중 가장 점수에 유리한 공격을 하게끔 학습된다. 그러나 어떤 경우에는 공격보다 이동이 더 좋은 선택이 될 수 있다. 예를 들어, 그림 6(a)와 같이 상대방 캐릭터가 필살기(s) 공격의 2단계를 수행하고 있을 때, 지능 캐릭터가 반드시 전진(f)을 행동해야만 필살기(s)의 유효거리를 벗어나서 주먹 공격을 할 수 있고, 그 외의 모든 공격 행동은 상대방 캐릭터의 공격을 받아 무효화되는 상황을 가정하자**. 학습

* 본 논문에서 사용한 게임 규칙에서 이동 행동은 무효화되지 않는다.
** 여기서, 전진(f)과 후진(b)은 행동하는데 소요되는 시간을 1 클럭으로 가정하고, 필살기(s)의 소요시간은

과정에서 지능 캐릭터가 전진이나 후진을 하면 점수를 획득하지 못하므로 같은 강화값 0으로 학습하게 된다. 따라서 실제 게임의 같은 상황에서는 그림 6(b)와 같이 전진을 하는 것이 아니라, 후진을 할 수도 있다. 이렇게 되면 지능 캐릭터는 필살기 공격의 유효거리를 벗어나지 못해 오히려 상대방 캐릭터가 점수를 획득하게 된다. 이와 같은 이유로 이동 행동을 학습하는 방법을 고안하였다.

본 논문에서 제안한 이동 행동에 대한 학습 방법은 그림 7과 같다. 만약 지능 캐릭터가 이동 행동을 한 후 다음에 공격 행동을 연속적으로 하여 점수를 획득하게 되면 먼저 수행한 이동 행동에 대해서는 강화값의 1/2*을 부여하여 학습하는 것이다. 예를 들어 지능 캐릭터가 전진을 한 후 발공격을 해서 점수를 획득하고 이때 강화값이 5라면 이전 행동인 전진에 대해서도 강화값을 2.5로 설정하여 강화 학습한다. 여기서 이동 행동에 대해 강화값의 1/2을 사용하는 이유는 과거의 이동은 현재의 공격 행동 보다는 점수에 영향을 적게 주었다고 생각할 수 있기 때문이다.

이 방식을 확장하여 연속적인 이동 행동 뒤에 공격 행동을 했을 경우, 앞서 수행한 이동 행동들에도 같은 방식을 적용할 수 있다. 예를 들어 이동 행동을 2번하고, 공격 행동을 해서 점수를 획득했을 경우 마지막 이동에 대해서는 공격 행동의 강화값의 1/2을 사용하고, 그 이전 이동 행동에 대해서는 공격 행동의 강화값의 1/4을 적용할 수 있다.

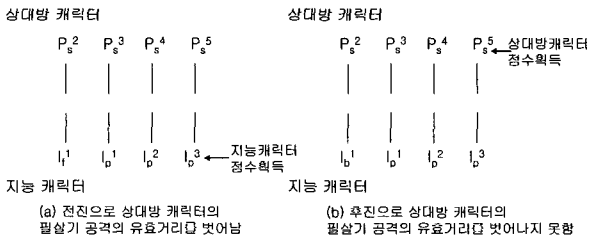


그림 6. 이동 행동으로 공격의 유효거리를 벗어나는 경우와 벗어나지 못하는 경우

Fig. 6. Moving action getting out of range of attack and vice versa.

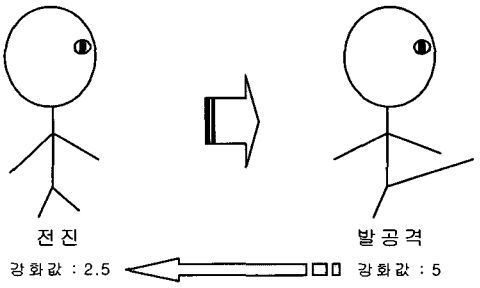


그림 7. 이동 행동에 대한 강화값

Fig. 7. Reinforcement value of moving action.

Ⅳ. 실 험

본 논문에서 제안한 지능 캐릭터를 검증하기 위하여 간단한 대전 액션 게임 모델을 개발했다. 게임은 두 캐릭터가 제한된 1차원 게임 공간에서 이동하면서 공격과 방어를 하며, 그 결과에 따라 점수를 획득한다. 캐릭터가 할 수 있는 행동은 표 1과 같고, 캐릭터의 모든 행동은 클럭에 동기가 맞춰져 있다. 각각의 공격 행동을 하는데 소요되는 시간을 표 2에 나타냈고, 그 외 행동은 모두 1 클럭이다.

캐릭터가 할 수 있는 공격은 크게 주먹공격, 발공격 그리고 먼 거리에서 행할 수 있는 필살기의 세 종류로 구분되며, 주먹공격과 발공격은 각각 상대방 캐릭터의 하반신을 공격하는 아래공격과 상반신을 공격하는 위공격으로 세분화된다. 실제 게임처럼 각각의 공격에 따라 공격의 효과를 얻을 수 있는 유효거리와 공격점수를 설정하였다. 그리고, 아래주먹과 아래발공격은 상대방이 막거나 점프를 하면 점수를 얻지 못하고, 위주먹과 위발공격은 상대방이 막거나 앉으면 점수를 얻지 못한다. 필살기 공격은 상대방이 막으면 50%의 점수만을 획득한다.

본 논문에서는 위에서 설명한 대전 액션 게임을 이용하여 지능 캐릭터가 상대방 캐릭터의 행동 패턴을 학습하여 적절히 행동하는지 여부를 알아보았다. 이것을 알아보기 위해서는 상대방 캐릭터가 어떤 방식으로 행동했을 때 지능 캐릭터의 학습 여부를 효과적으로 알아볼 수 있는지에 대한 고찰이 필요하며 다음은 이에 대한 내용이다.

본 논문에서 제안한 지능 캐릭터가 상대방 캐릭터의

표 2. 각 공격의 소요시간/공격점수/유효거리

Table 2. Necessary time/Attack score/Effective range.

공격행동	소요시간(클럭)	공격점수	유효거리
아래주먹	2	1	0~2
위주먹	4	2	
아래발	6	3	2~3
위발	8	4	
필살기	10	5	3~5

5, 주먹공격(p)의 소요시간은 3으로 가정한다.

* 0과 1 사이의 값을 사용하면 되나, 몇 차례의 사전 실험에서 우수한 성능을 보인 1/2을 사용한다.

표 3. 실험에 사용한 행동 패턴과 각 패턴에 대한 지능 캐릭터의 최적 행동

Table 3. Experimental action patterns and the optimal actions of intelligent characters against each pattern.

번호	상대방 캐릭터의 행동 패턴	지능 캐릭터의 최적 행동	캐릭터간의 초기거리	점수
1	전진,위주먹,후진,아래발,후진,필살기,전진,위발	위주먹,아래발,필살기,위발	3	14
2	전진,위주먹,후진,아래발,전진,아래주먹,후진,위발	위주먹,아래발,아래주먹,위발		10
3	후진,필살기,전진,아래발,후진,필살기,전진,위발	필살기,아래발,필살기,위발		17
4	아래주먹,아래주먹,후진,아래발,아래발,전진,아래주먹,아래주먹	후진,필살기,필살기	2	10

행동 패턴을 학습하여 대응하는지 알아보기 위해 실험에 사용한 행동 패턴은 표 3에 나타낸 네 가지이다. 표 3의 실험에 사용한 행동 패턴에서 패턴 4는 3.2절에서 설명한 이동 행동에 대한 학습을 잘 하는지 알아보기 위한 패턴이다. 왜냐하면 행동 패턴 4는 실험시 주어진 상황에서 상대방 캐릭터가 공격할 수 있는 가장 짧은 공격만을 하도록 구성한 행동 패턴이기 때문이다. 3.2절에서도 설명한 것처럼 상대방 캐릭터가 어떤 상황에서 가장 짧은 공격만을 한다면 지능 캐릭터가 취할 수 있는 것은 같은 공격을 하든지 (앞에서도 언급한 것처럼 이 경우에는 점수차가 0이 되어 학습이 되지 않는다.) 이동을 통하여 상대방 공격의 유효범위를 벗어날 수 있으면 이동을 하는 것이 좋다. 이동 행동에 대한 학습 방법이 없는 기존의 방법에서는 행동 패턴 4에 대한 대응을 할 수가 없으나 본 논문에서 제안한 이동 행동에 대한 학습이 가능한 방법에서는 가능하다 (실제로 4절의 실험 결과를 보면 행동 패턴 4에 대해서 이전의 방법은 거의 점수를 획득하지 못하나 본 논문에서 제안한 지능 캐릭터는 높은 점수를 획득하는 것을 볼 수 있다.).

표 3에서 지능 캐릭터의 최적 행동과 점수는 상대방 캐릭터가 해당 행동 패턴을 반복했을 때 지능 캐릭터가 가장 많은 점수를 획득할 수 있는 행동과 그 때 획득할 수 있는 점수를 표시한 것이다.

예를 들어 상대방 캐릭터가 행동 패턴 2의 순서대로 행동하고, 그 때 지능 캐릭터가 최적 행동을 하면 표 4와 같이 점수가 발생한다. 표 4의 항목 중에서 "구간"은 설명을 쉽게 하기 위해서 두 캐릭터의 행동 시작부터 점수가 발생할 때까지를 하나로 묶어서 표시한 것이다. 먼저 1 구간을 보면 상대방 캐릭터가 전진을 할 때 지능 캐릭터는 위주먹으로 공격한다. 상대방 캐릭터가 전진 후 위주먹으로 공격하지만, 지능 캐릭터의 위주먹이 먼저 종료하기 때문에 지능 캐릭터가 2점을 획득하고 상대방 캐릭터의 위주먹의 4단계는 무효화된다. 상대방 캐릭터의 공격은 무효화되었고, 지능 캐릭터의 행동은 종료했기 때문에 두 캐릭터 모두 새로운 행동을 시작한다. 구간 2에서 상대방 캐릭터는 행동 패턴대로 후진을 하고, 그 때 지능 캐릭터는 아래발로 공격한다. 구간 1과 같은 이유로 지능 캐릭터가 3점을 획득하고 상대방 캐릭터의 아래발의 6단계는 무효화된다. 같은 방식으로 구간 3, 4까지 종료하면 상대방 캐릭터의 행동 패턴 2가 종료하고, 다시 행동 패턴을 반복하게 된다.

표 4. 행동 패턴 2에 대한 최적 행동 (두 캐릭터 간의 초기 거리: 3 P: 상대방 캐릭터 N: 지능 캐릭터 f:전진, dp:아래주먹, up:위주먹, dk:아래발 uk:위발)

Table 4. The optimal actions for action pattern 2.

구간	거리	2	3	2	3	2
1	P	f1 up1 up2 up3	<-- N의 공격을 받아 up4 무효화			
	N	up1 up2 up3 up4	2점 획득			
2	P		b1 dk1 dk2 dk3 dk4 dk5		<-- N의 공격을 받아 dk6 무효화	
	N		dk1 dk2 dk3 dk4 dk5 dk6		3점 획득	
3	P		f1 dp1		<-- N의 공격을 받아 dp2 무효화	
	N		dp1 dp2		1점 획득	
4	P		b1 uk1 uk2 uk3 uk4 uk5 uk6 uk7		<-- N의 공격을 받아 uk8 무효화	
	N		uk1 uk2 uk3 uk4 uk5 uk6 uk7 uk8		4점 획득	
5	P				f1 up1 up2 up3	
	N				up1 up2 up3 up4	
		<-- 행동 패턴 시작			<-- 행동 패턴 반복	

논문 [10]에서 제안한 지능 캐릭터*와 본 논문에서

* 본 논문에서 제안하는 지능 캐릭터에서 상대방 캐릭터의 과거 행동을 입력받지 않고, 이동 행동에 대한 학습을 하지 않는 지능 캐릭터

표 5. 여러 행동 패턴에 대한 실험 결과
(총 실험 수 : 10회)

Table 5. Experimental results against action patterns.
(the number of experiments : 10)

	행동 패턴 1		행동 패턴 2		행동 패턴 3		행동 패턴 4	
	최적 학습 횟수	학습시간 평균	최적 학습 횟수	학습시간 평균	최적 학습 횟수	학습시간 평균	최적 학습 횟수	학습시간 평균
단순지능 캐릭터	0	NA	0	NA	0	NA	0	NA
패턴학습 지능캐릭터	7	390000	10	150000	8	390000	5	550000

제안한 지능 캐릭터를 이용하여 실험하고 결과를 비교하였다. 편의상 첫 번째는 단순 지능 캐릭터, 두 번째는 패턴 학습 지능 캐릭터로 표시한다. 두 캐릭터간의 초기 거리는 행동 패턴 1,2,3은 3으로 설정한다. 행동 패턴 4의 초기 거리는 2로 설정하여 상대방 캐릭터가 주먹과 발 공격의 유효거리를 왕복하면서 짧은 시간을 소요하는 공격을 반복하도록 한다. 실험은 상대방 캐릭터가 각각의 행동 패턴을 반복적으로 수행하며, 지능 캐릭터는 10000 클럭 단위로 최대 100만 클럭까지 학습한 후 지능 캐릭터의 학습 결과를 살펴보았다. 신경망의 은닉 노드는 30개, 과거 행동 입력의 개수는 3개, 학습률은 0.5로 하여 실험하였다. 각 실험은 10개의 무작위 초기값을 사용하여 실험하였다. 각각의 행동 패턴에 대하여 10개의 무작위 초기값 중 몇 가지 경우에 대하여 최적 행동을 학습했는지와 그 때의 학습 시간에 대하여 살펴보았다. 그 결과를 표 5에 나타냈다.

표 5는 두 가지 지능 캐릭터가 네 가지 행동 패턴에 대하여 학습한 실험 결과를 보여준다. 표 5에서 보듯이 단순 지능 캐릭터는 모든 행동 패턴에 대해 최적 행동을 학습하지 못하였다. 반면에 패턴 학습 지능 캐릭터는 많은 경우에 있어서 최적 행동을 찾아내었다. 패턴 학습 지능 캐릭터가 모든 경우에 최적 행동을 찾지 못하는 이유는 학습 시간의 부족으로 최적 행동을 찾아내지 못했거나 혹은 신경망 학습에 있어서 지역 최적해에 빠졌기 때문인 것으로 판단된다. 그렇지만 단순 지능 캐릭터에 비하여 월등히 최적 행동을 학습하는 것을 볼 수 있었다.

그림 8~11은 두 지능 캐릭터의 10 가지 무작위 초기값에 대한 점수비의 평균을 보여준다. 이 점수비는 각 학습 단계에서 상대방 캐릭터가 행동 패턴을 일정 회수

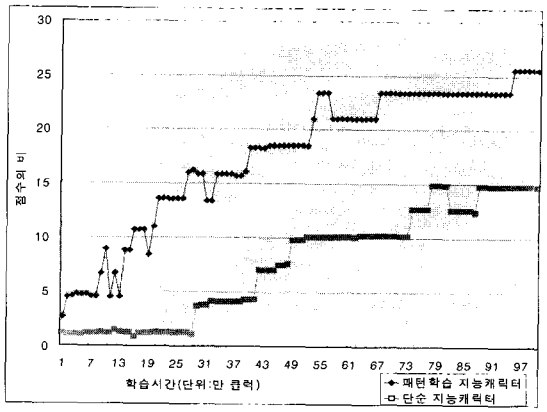


그림 8. 행동 패턴 1의 단순 지능 캐릭터와 패턴 학습 지능 캐릭터의 점수비

Fig. 8. The score ratio of simple and pattern learning intelligent characters against action pattern 1.

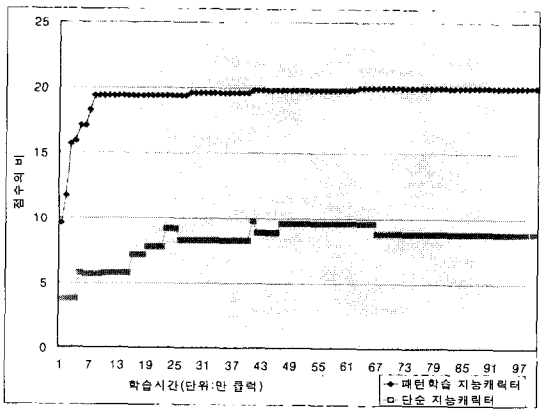


그림 9. 행동 패턴 2의 단순 지능 캐릭터와 패턴 학습 지능 캐릭터의 점수비

Fig. 9. The score ratio of simple and pattern learning intelligent characters against action pattern 2.

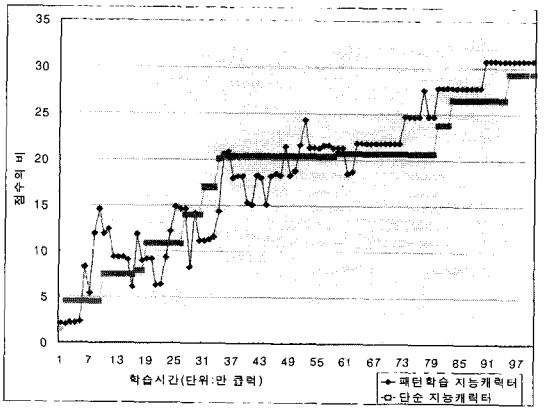


그림 10. 행동 패턴 3의 단순 지능 캐릭터와 패턴 학습 지능 캐릭터의 점수비

Fig. 10. The score ratio of simple and pattern learning intelligent characters against action pattern 3.

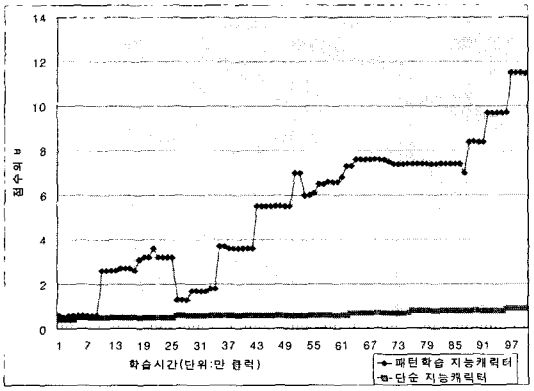


그림 11. 행동 패턴 4의 단순 지능 캐릭터와 패턴 학습 지능 캐릭터의 점수비

Fig. 11. The score ratio of simple and pattern learning intelligent characters against action pattern 4.

반복할 동안 두 캐릭터가 획득한 점수비이다. 행동 패턴 1, 2, 4에 대해서는 패턴 학습 지능 캐릭터가 학습 초기에는 물론 학습을 오래할수록 단순 지능 캐릭터보다 최적 행동을 많이 학습하여 높은 점수비를 보인다. 이와 같이 패턴 학습 지능 캐릭터가 단순 지능 캐릭터보다 점수비가 높게 나오는 이유는 다음과 같다. 3.1에서 설명했듯이, 상대방 캐릭터의 과거 행동을 입력받기 때문에 상대방 캐릭터가 같은 행동을 했을 때, 단순 지능 캐릭터와는 달리 상대방 캐릭터의 과거 행동에 따라 더 나은 행동을 학습할 수 있기 때문이다. 반면에 행동 패턴 3에 대해서는 학습을 많이 하면 패턴 학습 지능 캐릭터가 약간 점수비가 높지만, 두 지능 캐릭터의 점수비가 비슷한 추세로 증가한다. 이러한 결과가 나오는 원인은 두 가지이다. 첫 번째는 행동 패턴 3은 다른 행동 패턴과는 달리 상대적으로 소요시간이 큰 공격들(필살기,위발,아래발)을 포함하고 있기 때문이다. 이것은 상대방 캐릭터의 공격이 종료하기 전에 지능 캐릭터가 먼저 종료할 수 있는 공격의 종류가 많다는 것을 의미한다. 그러므로 두 가지 지능 캐릭터는 학습을 오래 하지 않은 중간 과정에서도 점수를 많이 획득할 수 있다. 두 번째 원인은 단순 지능 캐릭터가 비록 최적 행동을 학습하지는 못하지만, 학습한 행동으로 획득하는 점수가 패턴 학습 지능 캐릭터가 획득하는 점수와 큰 차이가 없기 때문이다.

표 6에 나타낸 바와 같이 행동 패턴 3에 대해서 학습한 결과를 이용하여 상대방 캐릭터와 대결하면 단순 지능 캐릭터는 16점, 패턴 학습 지능 캐릭터는 17점을 획득한다. 그리고 행동 패턴 4의 실험 결과에서 단순 지능 캐릭터는 거의 0점을 획득하는데, 그 이유는 두 가지이다. 첫 번째는 단순 지능 캐릭터는 이동 행동을 학습하지 못하기 때문이고, 두 번째는 상대방 캐릭터가 이동하면서 그 위치에서 할 수 있는 가장 짧은 공격을 반복하기 때문이다. 이와 같은 이유로 단순 지능 캐릭터가 학습 도중에 상대방 캐릭터보다 점수를 많이 획득하는 공격 행동을 학습할 기회가 적게 된다. 결과적으로 단순 지능 캐릭터가 점수를 획득하기도 하지만, 평균적으로는 거의 0점을 획득한다.

표 6. 행동 패턴 3의 학습 결과

Table 6. Result of learning for action pattern 3.

	학습행동	획득하는 점수
단순 지능 캐릭터	필살기,아래발,필살기,아래발	16
패턴 학습 지능 캐릭터	필살기,아래발,필살기,위발	17

표 7. 무작위로 행동하는 상대방 캐릭터로 학습한 실험 결과

Table 7. Experimental result of learning against random action characters.

	단순 지능 캐릭터		패턴 학습 지능 캐릭터	
학습시간	평균점수비	표준편차	평균점수비	표준편차
1000	0.1	0.0	0.3	0.0
2000	0.0	0.0	0.5	0.0
4000	0.0	0.0	0.9	0.1
8000	0.8	0.3	1.4	0.2
16000	1.8	0.2	1.6	0.1
32000	1.9	0.1	1.5	0.1
64000	2.2	0.1	1.5	0.0
128000	2.3	0.0	2.1	0.1
256000	2.4	0.0	2.8	0.0
512000	2.8	0.2	3.2	0.0
1024000	3.1	0.2	3.3	0.1
2048000	3.4	0.2	3.5	0.1
4096000	3.6	0.1	3.8	0.0
8192000	3.6	0.3	4.0	0.0

본 논문에서 제안한 방법이 특정 행동 패턴을 보이는 상대방 캐릭터에게만 효과적인지 알아보기 위하여 무작위로 행동하는 상대방 캐릭터와 실험하여 보았다.

표 7에서 보듯이, 패턴 학습 지능 캐릭터가 학습시간이 증가함에 따라 단순 지능 캐릭터보다 소폭 성능이 높아지는 것을 알 수 있다. 이와 같이 소폭 높아지는 이유는 상대방 캐릭터가 규칙적이지 않고 무작위로 행동하기 때문에 지능 캐릭터가 보다 나은 결정할 할 기회가 적기 때문이다. 지금까지 살펴본 실험 결과로써 본 논문에서 제안한 지능 캐릭터는 상대방 캐릭터가 특정 행동을 반복하는 경우는 물론, 무작위로 행동하는 경우에도 더 나은 성능을 보인다는 것을 보여준다. 그러므로 논문 [10]에서의 지능 캐릭터보다 더 효과적인 구현 알고리즘이라고 판단된다.

V. 결 론

본 논문에서는 신경망을 이용하여 일반적인 대전 액션 게임에서 상대방 캐릭터의 행동 패턴을 학습하는 지능 캐릭터를 구현하는 방법을 제안하였다. 또한 공격이나 방어 행동과는 달리 행동의 적절성을 판단하기 어려운 이동 행동에 대한 학습 방법도 제안하였다. 제안한 알고리즘은 지능 캐릭터가 행동을 결정하는 시점에 상대방 캐릭터의 과거 행동을 포함하는 입출력 값과 상대방 캐릭터와 지능 캐릭터 사이의 점수의 차를 이용하여 강화 학습하였다. 제안한 방법을 테스트해 보기 위하여 실제와 유사한 대전 액션 게임을 개발 적용했고, 상대방 캐릭터의 네 가지 행동 패턴에 대하여 실험하였다. 실험 결과 모든 행동 패턴에 대하여 지능 캐릭터가 최적 행동을 학습했다. 그리고 여러 가지 무작위 초기값에 대하여 우수한 평균 점수비를 보였고, 또한 무작위로 행동하는 캐릭터에 대해서도 지능적으로 대처함을 보았다. 이와 같은 결과로써 지능 캐릭터가 게임의 규칙과 상대방 캐릭터의 행동 패턴을 학습하여 상대방 캐릭터에 대해 적절히 대응한다는 것을 알 수 있다. 본 논문에서 제안한 방법은 비록 신경망의 입력 개수를 늘어나게 하여 복잡도를 증가시키고, 학습 시간이 길어지는 단점이 있다. 그러나 상대방 캐릭터의 행동 경향을 파악하여 행동을 할 수 있게 학습으로써 보다 인간과 유사한 지능 캐릭터를 구현할 수 있게 한다. 본 논문에서 제안한 지능 캐릭터는 대전 액션 게임을 위한 것이기 때문에, 캐릭터들이 행동을 함으로써 결과가 나타나는 게임들, 예를 들어 전략 게임이나 최근 많이 개발되어 서비스되고 있는 온라인 게임 등에 적용될 수 있어서 활용할 수 있는 곳이 매우 많다. 온라인 게임과 같이 동일 종족의 개체가 다수 분포하는 경우에는 지능 캐릭터가 외부 환경 변화에 적응할 필요성이 있다. 추후에 이와 관련된 연구를 수행할 예정이다,

참 고 문 헌

[1] Daniel Fu, Ryan Houlette, Stottler Henke, "Putting AI In Entertainment: An AI Authoring Tool for Simulation and Games," IEEE Intelli-gent and Systems July/August, Vol.17, No.4, 2002.

[2] Daniel Johnson, Janet Wiles, "Computer Games With Intelligence," IEEE International Fuzzy Systems Conference, 2001.

[3] Mark DeLoura, Game Programming Gems 2, Charles River Media, 2001.

[4] Bernd Freisleben, "A Neural Network that Learns to Play Five-in-a-Row," 2nd New Zealand Two-Stream International Conference on Artificial Neural Networks and Expert Systems, 1995.

[5] David B. Fogel, "Using Evolutionary Program-ming to Create Neural Networks that are Capable of Playing Tic-Tac-Toe," Intl. Joint Confrence Neural Networks, New York, pp.875-880, 1993.

[6] 조남덕, 성백균, 김기태, "인공생명 시뮬레이션을 통한 게임 캐릭터의 전략 구현," 정보과학회 2000년 춘계학술대회, VOL.27, NO.01, pp.0241~0243, April, 2000.

[7] 이만재, 게임에서의 인공지능 기술, 한국정보처리학회지, Vol. 9, No. 3, pp.69-76, May, 2002.

[8] Steve Rabin, AI Game Programming Wisdom, Charles Rivers Media, 2002.

[9] 조병헌, 정성훈, 성영락, 오하령, "신경망을 이용한 지능형 게임 캐릭터의 구현," 정보처리학회, submitted for publication 2004.

[10] 조병헌, 정성훈, 성영락, 오하령, "대전 액션 게임을 위한 신경망 지능 캐릭터의 구현," 한국 퍼지 및 지능시스템학회 논문지 Vol. 14, No. 4, pp.383~389, July, 2004.

[11] Chin-Teng Lin, C. S. George Lee, Neural Fuzzy Systems, Prentice Hall, 1996.

[12] Richard. P. Lippman, An Introduction to Compu-ting with Neural Nets, IEEE ASSP Magazine, pp.4-22, April, 1987.

저 자 소 개

조 병 헌(학생회원)

1997년 국민대학교 전자공학과 (공학사)
1999년 국민대학교 전자공학과 (공학석사)
1999년~현재 국민대학교 전자정보통신공학부 박사과정

<주관심분야: 유전자 알고리즘, 시뮬레이션, 신경망>

성 영 락(정회원)

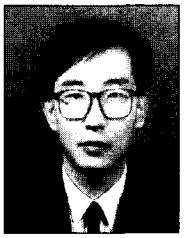
1989년 한양대학교 전자공학과 (공학사)
1991년 한국과학기술원 전기 및 전자공학과(공학석사)
1995년 한국과학기술원 전기 및 전자공학과(공학박사)
1995년~1996년 한국과학기술원 위촉연구원
1996년~1998년 국민대학교 전자공학부 전임강사
1998년~2002 국민대학교 전자공학부 조교수
2002년~현재 국민대학교 전자정보통신공학부 부교수

<주관심분야: 시뮬레이션, 고장감내, 내장형 시스템>

정 성 훈(정회원)

1988년 한양대학교 전자공학과 (공학사)
1991년 한국과학기술원 전기및 전자공학과(공학석사)
1995년 한국과학기술원 전기및 전자공학과(공학박사)
1995년~1996년 한국과학기술원 전기및전자공학과(위촉연구원)
1996년~1998년 한성대학교 정보전산학부 정보통신공학전공 전임강사
1998년~2002년 한성대학교 정보전산학부 정보통신공학전공 조교수
2002년~현재 한성대학교 정보공학부 정보통신공학전공 부교수

<주관심분야: 지능 시스템, 유전자 알고리즘, 신경망, 뉴로퍼지>



오 하 령(정회원)

1983년 서울대학교 전기공학과 (공학사)
1983년~1986년 삼성전자 종합연구소
1988년 한국과학기술원 전기전자과 컴퓨터공학전공(공학석사)
1992년 한국과학기술원 전기전자과 컴퓨터공학전공(공학박사)
1992년~1996년 국민대학교 전자공학부 조교수
1996년~2001 국민대학교 전자공학부 부교수
2001년~현재 국민대학교 전자정보통신공학부 교수

<주관심분야: 병렬처리, 내장형 시스템, 고장감내>