Journal of Korea Multimedia Society Vol. 11, No. 9, September 2008(pp. 1258-1266)

적응형 NPC 생성을 위한 FSM의 동적 활용 방안

양정모[†], 조경은[††], 엄기현[†††]

요 약

대개의 게임 플레이어들은 정해진 패턴대로만 행동하는 NPC(Non-Player Character)보다 사람 플레이어와 상호작용할 때 더욱 큰 만족을 얻는다. 하지만 항상 사람 플레이어를 상대할 수 있는 것은 아니기 때문에 다양한 게임 플레이어에 맞추어 행동할 수 있는 적응형 NPC가 필요하다. 본 논문에서는 FSM(Finite State Machine)을 이용하여 적응형 NPC를 생성하는 기법을 제안한다. 이 기법은 행동 데이터베이스의 행동 정보를 이용하여 FSM을 구성하고, 게임을 진행하는 동안 FSM의 종합 효율값이 목표 효율값으로 접근하도록 실시간으로 FSM을 갱신하는 것을 반복한다. 이 과정에서 NPC는 플레이어에게 적응한다. 실험은, 제안한 기법을 적용한 2D 게임을 제작하여 목표 효율값을 다양하게 설정하여 진행하였다. 실험 결과로, 게임이 진행되면서 종합 효율값이 목표 효율값으로 접근하는 것을 볼 수 있었다. 이는 NPC가 플레이어에게 적응하여 적응형 NPC가 생성됨을 의미하는 것이다.

A Dynamic Utilization method of FSM for Adaptive NPC Generation

Jeongmo Yang[†], Kyungeun Cho[††], Kyhyun Um[†††]

ABSTRACT

Most game players obtain more satisfactions by interacting with human players that have fluxed behavior patterns, than with NPC(Non-Player Character)s that have fixed behavior patterns. Since it is impossible that game players always interact with human players, adaptive NPCs that can variously behave are required. In this paper, we present a method to create adaptive NPCs using a dynamic FSM(Finite State Machine). This method configures a dynamic FSM by using behavior information at behavior database, and repeatedly updates the dynamic FSM so that the dynamic FSM's total efficiency approaches to a given target efficiency. NPC adapts to game players through this process. For an experiment, we have implemented a 2D game with this strategy, and experimented with various target efficiencies. We show that a dynamic FSM's total efficiency approaches to target efficiency by updating a dynamic FSM several times over. It means that the adaptive NPC to be generated, adapts to game players.

Key words: Adaptive NPC(적응형 NPC), Dynamic FSM(동적 FSM), Adaptation(적응)

1. 서 론

게임 플레이어가 게임을 하면서 가장 크게 기대하는 것은 만족이다[1]. 만족을 얻기 위해서, 게임 플레이어들은 게임에서 그래픽과 같이 겉으로 보이는 다양성외에도, 게임 속에서 게임 플레이어와 직접적으로 상호작용하는 NPC(Non-Player Character)의 다양성을 요구한다. 일반적인 게임들에 등장하는 NPC

※ 교신저자(Corresponding Author) : 조경은, 주소 : 서울시 중구 필동 3가 26(100-715), 전화 : 02)2260-3834, FAX : 02)2260-3766, E-mail : cke@dongguk.edu
접수일 : 2008년 1월 16일, 완료일 : 2008년 7월 28일
[†] 준회원, 동국대학교 컴퓨터공학과
(E-mail : l38472@naver.com)
[††] 종신회원, 동국대학교 영상미디어대학 게임멀티미디어공학과 조교수
[†††] 종신회원, 동국대학교 영상미디어대학 게임멀티미디어공학과 교수
(E-mail : khum@dongguk.edu)

는 항상 같은 정해진 패턴으로 행동한다. 게임 플레이어들 중 숙련자는 이런 NPC와의 상호작용에서 게임이 너무 쉬울 수도 있고, 초보자의 경우는 너무 어려울 수도 있다. 이에 적응(Adaptation)이라는 개념을 도입하여 게임 속에 존재하는 NPC를 게임 플레이어에게 적응시켜 행동을 변화시킬 수 있다[2]. 적응이란 사전적으로 생물이 주위 환경에 적합하도록 형태적, 생리학적으로 변화함, 또는 그런 과정을 뜻한다[3]. 이런 NPC를 적응형 NPC라고 한다.

게임에서 NPC가 더 지능적이고 자율적으로 행동하는 것과 더불어, 적응형 NPC를 생성하는 체계에 대한 연구가 활발하다. 본 논문에서는 적응형 NPC를 생성하기 위해 NPC가 실시간으로 게임 플레이어에게 적응하는 기법을 제시한다. 이 기법은, 실시간으로 관리되는 NPC 고유의 행동 데이터베이스(DB)를 이용하여 NPC의 행동을 제어하는 유한상태기계(Finite State Machine, FSM)을 동적으로 구성하는 방법을 사용한다. 이 FSM을 동적 FSM이라 한다. 이로써 NPC의 행동을 실시간으로 변화시킬 수 있고 그 결과 적응형 NPC를 생성할 수 있다. 적응형 NPC를 이용하면, 게임 플레이어들은 좀 더 만만한 NPC와 게임을 진행할 수 있어 더 큰 만족감을 기대할 수 있으며, 동적인 게임 밸런싱(game balancing)이 가능하다[4]. 나아가, 교육용 게임의 보조 교사, 훈련용 게임의 상대자, 쇼핑몰의 안내자 등으로 활용할 수 있다.

본 논문의 구성을 살펴보면, 2장에서는 기존의 적응형 NPC에 관한 주요 연구들을 분석하여 기존 연구와의 차이점을 설명한다. 3장에서는 적응형 NPC를 생성하기 위한 제어 구조와 그 구성 요소를 설명하며, NPC가 실시간으로 플레이어에게 적응하는 구체적인 방안을 설명한다. 4장에서는 제안된 방안을 간단한 2D 액션 게임에 적용하여 실험하고 그 결과를 분석한다. 마지막 5장에서는 본 연구의 결론을 서술한다.

2. 관련 연구

2.1 기존 연구

대개의 게임에서 NPC의 행동을 제어하는 데는 기본적으로 FSM을 많이 이용한다. 적응형 NPC를 생성하기 위한 기법으로는, 강화 학습에 기반한 방법[5,6], 행동 규칙에 기반한 방법[7], 행동 관찰 인식 모델의 다단계 학습을 이용한 방법[8], 유전자 알고리즘에 기반한 방법[9] 등이 제안되어 있다.

강화학습 기반 연구는, 현재의 게임 난이도에 따라 게임을 쉽게 하거나 어렵게 하는 액션을 NPC가 실행하게 하여 적응형 NPC를 생성하는 방법이다. 이 방법 중에는 신경망을 이용한 NPC 적응 기법이 있다. 이 기법에서 초기 NPC는 무작위로 행동을 선택하여 실행한다. 그 행동 선택의 결과로 얻은 게임 결과값에 의한 강화값을 이용해 신경망의 링크 가중치를 갱신한다. 초기 신경망의 링크 가중치는 무작위 값이므로 게임 플레이어의 행동에 대해 적절한 행동을 출력하지 못하지만, NPC의 신경망은 특정한 상황에 따라 어떤 행동이 적절한 행동인지를 학습하여 행동할 수 있게 된다[6].

행동 규칙 기반 연구의 대표적인 것이 동적 스크립팅(Dynamic Scripting)이다. 이 기법에서는 행동 규칙들이 저장된 NPC 고유의 규칙 베이스를 기반으로 스크립트를 생성하여 NPC의 행동을 제어한다. NPC와 게임 플레이어의 상호작용 결과를 토대로 규칙 베이스를 갱신한다. 이 후 다시 스크립트를 생성하여 NPC를 제어하는 과정을 반복하면서 NPC가 플레이어에게 적응해 간다.

이 기법은 상호작용 외의 시간에 필요한 연산을 수행한다. 빠르게 실행될 수 있고, 반복적인 규칙 베이스 갱신을 통해 효과적으로 플레이어에게 적응 가능하다. 하지만 게임이 복잡해질수록 규칙들을 유지 관리하는데 많은 어려움이 있고, 상호작용 중에는 스크립트를 갱신하지 않기 때문에 생성된 스크립트의 패턴으로만 행동해야 한다[7].

행동 관찰 인식 모델 기반 연구에서 행동 관찰 인식 모델의 다단계 학습을 이용한 방법으로 다단계 적응 기법이 있다. 이 기법에서 적응은 실시간 온라인 학습을 통해 사람 플레이어에 맞추어 전술을 조정하여 NPC를 강하게 만드는 것으로 정의한다. 캐릭터는 액션(action) 선택, 작업(task) 선택, 목표 선택의 단계로 구성된 행동 모델에 따라 적절한 액션을 선택하고 실행한다. 각 단계별로 사람 플레이어의 행동을 적절하게 예측하는 것과 어떤 상황에서도 가장 좋은 행동을 선택하도록 학습한다. 이 학습은 사람 플레이어의 행동을 모방하고, 캐릭터의 행복 수치를 최대화하는 감성 피드백을 통해 이루어진다. 캐릭터는 학습

결과를 바탕으로 사람 플레이어에게 적응한다.

이 기법은 관찰 학습을 통해 짧은 시간에 적응하고, 지식을 수집하는 것은 빠르지만, 지식을 사용할 때 상당히 많은 프로세서 자원을 소모하는 부담이 있다[8].

유전자 알고리즘 기반 연구에서, 유전자 알고리즘은 플레이어의 습성 또는 행동 양식이 복잡하고 다양하기 때문에 탐색 공간이 매우 큰 게임의 경우 적용하기 유리하다. 게임의 특성상 최적의 해만을 요구하는 것은 아니다. 적응 방법은 초기 게임의 집합인 세대를 구성하고, 유전자 세대를 적용한 게임의 적합도를 평가한다. 적합도 함수를 이용해 사용자의 수준을 판단하며 이에 따라 선택, 교배, 돌연변이를 수행하여 새로운 세대를 구성한다. 이 과정을 반복함으로써 플레이어 수준에 알맞은 세대를 구성하게 되고, 그 결과 플레이어에게 적응하여 알맞은 게임 난이도를 제공할 수 있게 된다[9].

2.2 기존 연구와 본 연구의 차이점

기존 연구들은 NPC의 행동 제어에 스크립트나 특정한 행동 모델을 사용함으로써 NPC의 행동을 어떻게 제어하느냐에 따라 게임에 적용하기가 제한적이거나 어렵다. 본 논문에서 제안하는 기법은 FSM을 사용하여 NPC를 제어한다. 이 기법은 FSM 상태 교체 시에 간단한 알고리즘을 적용하여 프로세서 자원을 적게 소모한다. 또한 동적 스크립팅과는 달리 상호작용 중에도 FSM 상태를 주기적으로 실시간 갱신한다. 그래서, 상호작용 중에도 NPC의 행동을 실시간으로 변화시킬 수 있다. 행동 DB는 사람 플레이어의 행동을 관찰하여 실시간으로 행동 정보를 저장한다. 그 정보들을 이용하여 다양한 FSM을 구성할 수 있다. 그 결과 다양하게 행동하는 NPC를 생성하는 것이 가능하다.

3. 동적 FSM을 이용한 적응 기법

게임 속에서 NPC가 게임 플레이어에게 적응하기 위해서는 NPC의 행동이 플레이어에 따라 적절하게 변화해야 한다. 게임 플레이어의 수준을 측정한 결과와 여러 가지 행동 정보를 바탕으로 실시간으로 FSM을 변경하면 NPC의 행동에 변화를 줄 수 있다. 이 장에서는 동적 FSM을 이용한 NPC 제어 구조와 적응 기법을 설명한다.

3.1 동적 FSM의 개념

게임에서 NPC의 행동을 제어하기 위해 일반적으로 FSM을 많이 사용한다. 이를 필요할 때마다 갱신하여 동적 FSM을 구성한다. 이를 [정의 1]처럼 정의하고, 이를 관리하기 위해 필요한 개념을 설명한다. 이것은 비교적 구현이 용이하다는 장점이 있다. [정의 1]에서, 상태를 변화시킨다는 것은 FSM의 상태를 갱신하거나 삭제하고, 새로운 상태를 추가하는 것을 말한다. 동적 FSM을 사용하면, 게임 플레이어에 맞추어 행동하면서 변경되는 상태 정보를 표현하는 것이 가능하다.

[정의 1] 동적 FSM (Dynamic FSM)이란, 게임을 실행하고 있는 중에도 실시간으로 상태를 변화시킬 수 있는 FSM을 말한다.

[정의 2] 동적 FSM 주기란, 동적 FSM이 처음 생성된 후 또는 갱신되고 난 후부터 다음 갱신 전까지의 기간을 말한다.

[정의 3] 종합 효율값(Total Efficiency)이란, 동적 FSM 한 주기 동안 실행되는 모든 행동들의 효율값 평균을 말한다. 이 값을 산출하는 식은 다음 식 (1)과 같다.

$$TE = \frac{\sum_{t=0}^{N-1} E(S_t, A - List_t)}{N}, N: (S, A - List)\text{개수} \quad (1)$$

[정의 4] 목표 효율값(Target Efficiency, 적응 목표값)이란, 종합 효율값이 변하여 최종 접근 목표가 되는 값이다.

종합 효율값은 현재 NPC가 게임 플레이어에게 얼마나 효율적으로 대응했는가를 나타내는 값이다. 값이 클수록 현재 게임 플레이어에 대해 강한 NPC이고, 값이 작을수록 현재 게임 플레이어에게 약한 NPC인 것을 나타낸다. 이 값은 현재 동적 FSM이 운영되고 있는 동안 실행된 모든 행동들의 효율값의 평균 값이다. 동적 FSM 한 주기에서 동적 FSM이 갱신될 때마다 산출된 값은 동적 FSM 갱신에 이용된다. 목표 효율값은 게임 플레이어에 대해 NPC의 강한 정도를 나타내는 값이다. 종합 효율값이 목표 효율값으로 접근하도록 NPC를 제어하면서 플레이

어에게 적응시킨다.

3.2 행동 데이터베이스

동적 FSM를 구성하기 위해 필요한 정보는 모두 NPC의 행동 DB에 저장한다. 게임에서 발생 가능한 상태 S, S에서 실행 가능한 액션들의 리스트 A-List, S에서 A-List를 실행했을 때의 효율값 E, 동적 FSM을 구성할 때 사용된 횟수 F, 그리고 FSM을 구성한 최근 시간 LST(Latest Selected Time)의 내용 등을 포함한다. 저장 형태는 표 1과 같다.

F값과 LST값은 행동 DB의 각 정보에 대한 필요 여부를 판단하는 근거가 된다. 효율값 E는 NPC와 게임 플레이어의 상호작용 결과 값과 게임의 환경값을 기반으로 게임 고유의 식에 의하여 산출된다. 동적 FSM과 행동 DB를 갱신할 때, 적절한 효율값 E를 가지고 있는 상태 S가 선택된다.

3.3 동적 FSM 상태 갱신

동적 FSM 내의 상태를 교체하는 알고리즘을 실행하여, 종합 효율값이 목표 효율값으로 접근시킨다. 이를 위해, 현재 동적 FSM에서 가장 낮은 효율값을 가진 상태를 삭제하고, 삭제한 상태보다 높은 효율값을 가지고 있는 상태를 동적 FSM에 추가한다. 반대로 종합 효율값이 목표 효율값보다 크면 종합 효율값을 작게 만든다. 이를 위해 현재 동적 FSM에서 가장 큰 효율값을 가진 상태를 삭제하고, 이보다 낮은 효율값을 가진 상태를 추가한다. 이를 서술한 것이 그림 1이다. 각 용어의 의미는 표 2와 같다.

이 작업을 반복함으로써 현재 동적 FSM의 종합 효율값이 결국 목표 효율값으로 접근하게 되고, 게임 플레이어가 원하는 적절한 수준의 적응형 NPC가 생성된다. 즉, NPC가 게임 플레이어에게 적응하여 목표 효율값에 알맞은 적절한 난이도를 제공하게 된다.

표 1. 행동 DB에 저장되는 정보 테이블

S	A-List	E	F	LST
게임에서 발생 가능한 상태	S 상태에서 실행 가능한 액션들 (1개 이상)	S 상태에서 A-List를 실행했을 때의 효율값	FSM을 구성할 때 선택되어 사용된 횟수	선택되어 FSM으로 구성된 최근 시간

```
If TE > GE Then
    If DecState 존재 Then
        FSM에서 FSMMaxState 삭제
        FSM에 DecState 추가
    Else If KBMinState 존재 Then
        FSM에서 FSMMaxState 삭제
        FSM에 KBMinState 추가
Else If TE < GE Then
    If IncState 존재 Then
        FSM에서 FSMMinState 삭제
        FSM에 IncState 추가
    Else If KBMaxState 존재 Then
        FSMMinState 삭제
        KBMaxState 추가
```

그림 1. 동적 FSM 상태 갱신 알고리즘

표 2. 동적 FSM 상태 갱신 알고리즘에 사용되는 각 용어의 의미

구 분	의 미
TE	종합 효율값 (Total Efficiency)
GE	목표 효율값 (Goal Efficiency)
IncState	현재 FSM에 존재하지 않고, 목표 효율값보다 크면서 그 중에서 가장 작은 효율값을 가진 State
DecState	현재 FSM에 존재하지 않고, 목표 효율값보다 작으면서 그 중에서 가장 큰 효율값을 가진 State
FSMMaxState	현재 FSM에서 가장 큰 효율값을 가진 State
FSMMinState	현재 FSM에서 가장 작은 효율값을 가진 State
KBMaxState	현재 FSM에 존재하지 않고 가장 큰 효율값을 갖는 State
KBMinState	현재 FSM에 존재하지 않고 가장 작은 효율값을 갖는 State

3.4 적응형 NPC 제어 구조

NPC 제어 구조는, 동적 FSM 모듈, 행동 DB 생성 모듈, 행동 DB 등으로 구성되어 있다. 동적 FSM 모듈은 행동 DB에 있는 정보를 기반으로 동적 FSM을 생성하고 NPC를 제어한다. 그동안 행동 DB 생성 모듈은 NPC와 플레이어의 상호작용 결과와 환경값을 이용하여 NPC의 행동 DB를 갱신한다. 이 때 기존 행동 정보의 효율값 정보나 새로운 행동 정보를 저장하며, 기존 행동 정보를 삭제하기도 한다. 이와 동시

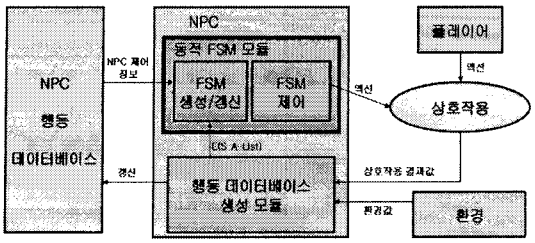


그림 2. 동적 FSM을 이용한 적응형 NPC 제어 구조

에, 현재 행동에 대한 효율값 정보는 당시 실행중인 동적 FSM의 갱신에 필요하기 때문에 FSM 생성 모듈에 전달된다. 그에 따라 동적 FSM의 기존 상태가 삭제되거나 새로운 상태가 추가되면서 동적 FSM이 갱신된다. 그림 2는 동적 FSM을 사용하는 NPC 제어 구조를 표현한 것이다.

4. 실험 및 분석

본 논문에서 제안한 NPC 적응 기법으로 생성된 NPC가 실시간으로 플레이어에게 제대로 적응하는지 여부를 실험한다. NPC가 플레이어에게 잘 적응하기 위해서는 동적 FSM의 지속적인 실시간 상태 갱신이 필요하다. 실험에서는 상태 갱신 정보들을 기록한 로그(log) 파일들을 관찰함으로써 NPC의 행동 패턴과 게임 플레이어의 행동 패턴을 비교 분석하고 목표 효율값에 따라 종합 효율값이 어떻게 변화하는지를 알아봄으로써 NPC가 적응하는지 여부를 판단한다.

4.1 실험 방법

본 실험의 가장 중요한 요소는 NPC가 다양하게 행동하는 것과, 행동하였을 때 게임에서 적절한 근거에 의해 실행한 행동의 효율값을 산출할 수 있는지 여부이다. 이 실험을 위해 제작한 게임에서 게임 플레이어와 NPC의 전투가 반복되는 동안 NPC의 행동 DB에 쌓이는 정보 로그 파일과 동적 FSM의 상태 로그 파일, 종합 효율값 변화 로그 파일, 각 캐릭터의 기술 사용 로그 파일 등 표 3의 로그 파일들이 생성된다.

실험은 두 가지로 진행된다. 실험 1은 초기에 NPC가 게임 플레이어에게 제대로 적응하는지 알아보는 것으로, 다양한 목표 효율값(100, 80, 60)을 설정하여 동적 FSM 100번째 주기가 될 때까지 진행한다. 실험

표 3. 게임 플레이시 생성되는 로그 파일의 종류

종류	생성 파일명	파일 내용 설명
행동 DB 관련 로그	DB_Log	행동 DB의 갱신 정보
	DB_BackUp	프로그램 종료시의 행동 DB의 모든 정보
	DB_Restore	프로그램 재시작시 DB 복구용 정보
동적 FSM 관련 로그	FSM_Log	동적 FSM 갱신 정보
	FSM_Log_backup	프로그램 종료시 동적 FSM 상태 정보
	FSM_Log_restore	프로그램 재시작시 동적 FSM 복구용 정보
	FSM_Log_TotalEfficiency	종합 효율값 변화 정보
캐릭터 기술 사용 로그	Skill_use_log_npc	NPC 행동 패턴 정보
	Skill_use_log_player	게임 플레이어 행동 패턴 정보

2는 게임 플레이어 A에게 적응한 NPC를 상대로 게임 플레이어 B가 게임을 했을 경우, NPC가 게임 플레이어 B에게도 적응하는지 여부를 알아보는 것이다. 이를 위해 게임 플레이어 A를 숙련자로 선정하여 동적 FSM 100번째 주기가 될 때까지 진행하고, 게임 플레이어 B는 초보자로 설정하여 마찬가지로 동적 FSM 100번째 주기가 될 때까지 진행한다. 총 200번째 동적 FSM 주기가 될 때까지 실험을 진행한다.

4.2 실험용 게임

실험용 게임은 플레이어가 키보드로 상하좌우로 이동하고 공격이 가능한 캐릭터가 NPC와 1대1로 대결하는 2D 액션 게임이다. 실험용 게임은 Windows XP 운영체제에서 Visual Studio .NET 2003, DirectX 9.0c 환경에서 구현한 것이다. 그림 3은 실험용 2D 액션 게임을 실행할 때의 게임 화면이다. 표 4는 구사할 수 있는 14 가지 기술을 정리한 것이다. 이 기술 중 최대 5개까지 조합하여 사용한다.

효율적으로 전투를 한다는 것은 적에게 데미지(damage)를 많이 주고, 자신은 적으로부터 데미지를 적게 받는 것을 의미한다. 이중에서도 적에게 받은 데미지보다 적에게 주는 데미지가 승리에 더 크게 영향을 준다. NPC가 적응하는 정도를 나타내는 효율값은, 적에게 주는 데미지 I의 1.5배에서 적에게 받는 데미지 D의 0.5배와 기술을 사용할 때 소모되는 체력

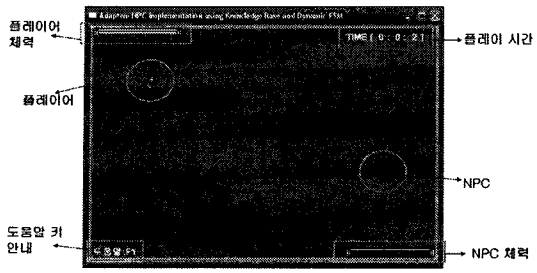


그림 3. 실험용 2D 액션 게임 화면 구성과 예

표 4. 실험용 게임에서 사용가능한 기술

기술명	키	데미지	효 과
일반공격	F	15	HP 소모 없음 - 특수 효과 없음
혼란	R	20	HP6소모-적의 방향을 반대로 돌림
밀쳐내기	A	70	HP2소모-적을 한 칸 뒤로 밀어냄
반격	S	-	HP6소모-적 공격력의 2배로 공격
뒤 잡기	D	20	HP4소모-적의 뒤로 빠르게 돌아감
회피	T	-	HP6소모-3초간 모든 공격 회피
도주	G	-	HP1소모-3초간 달리기 속도 증가
돌진	1	70	HP2소모-일정 거리를 달려가 공격
강베기	2	70	HP8소모-매우 큰 데미지
찌르기	3	20	HP6소모-일정 떨어진 거리에서 공격
연속베기	4	40	HP2소모-큰 데미지
진공베기	Q	40	HP2소모-일정 거리에서 검기 공격
기습	W	25	HP2소모-적 뒤에서 더 큰 데미지
기절	E	1	HP6소모-3초간 적을 행동불능시킴

S를 뺀 값을, 행동할 때 소요된 시간 T로 나누어 산출한다. 시간 단위는 초(sec)이다.

$$\text{효율값} E = \frac{(I \times 1.5) - \{S + (D \times 0.5)\}}{T} \quad (2)$$

4.3 실험 결과

실험 결과로 실험 중 수집한 데이터를 담은 9개의 로그 파일들이 생성된다. 본 연구의 분석 대상은 동적 FSM의 상태 변화를 기록한 FSM_Log, FSM_Log_TotalEfficiency, 그리고 각 캐릭터의 행동 패턴을 기록한 Skill_use_log_npc, Skill_use_log_player 등의 텍스트 파일이다.

표 5는 실험한 후 생성된 로그 파일 중 위 4개 파일이 갖고 있는 정보를 정리한 것이다. 실험 1에서 동적 FSM이 갱신될 때마다 기록하므로, 이 파일의 정보

표 5. 실험 후 생성된 로그 파일이 포함하고 있는 정보 통계

구 분		생성 파일 이름	정보건수
실험 1	목표 효율값 100	DB_Log.txt	1445
		FSM_Log_TotalEfficiency.txt	100
		Skill_use_log_npc.txt	380
		Skill_use_log_player.txt	504
	목표 효율값 80	DB_Log.txt	812
		FSM_Log_TotalEfficiency.txt	100
		Skill_use_log_npc.txt	1156
		Skill_use_log_player.txt	339
	목표 효율값 60	DB_Log.txt	974
		FSM_Log_TotalEfficiency.txt	100
		Skill_use_log_npc.txt	662
		Skill_use_log_player.txt	363
실험 2	숙련자 ↓ 초보자	DB_Log.txt	2866
		FSM_Log_TotalEfficiency.txt	200
		Skill_use_log_npc.txt	1151
		Skill_use_log_player.txt(A)	363
		Skill_use_log_player.txt(B)	550

건수는 100개로 일정하다. 실험 2에서도 종합 효율값의 정보 건수는 실험한 동적 FSM 주기와 같이 200개의 정보를 포함한다. 하지만 다른 정보들은, 게임 플레이어와 동적 FSM에 의해 행동하는 NPC에 따라 달라지므로 실험마다 일정하지 않은 결과가 나타났다.

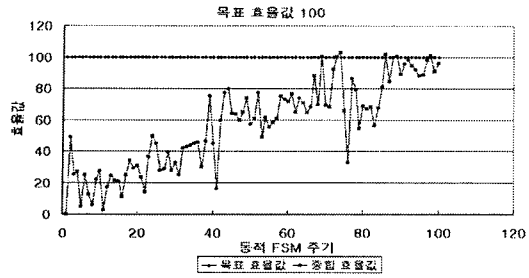
4.4 실험 결과 분석

4.4.1 실험 1 분석

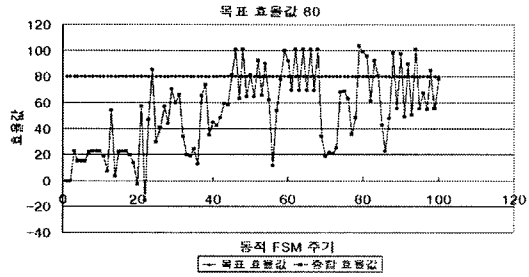
FSM_Log_TotalEfficiency 파일은 목표 효율값과 종합 효율값의 변화 정보를 포함한다. 먼저 각 실험 후 생성된 FSM_Log_TotalEfficiency 파일의 정보를 이용하여 종합 효율값이 목표 효율값에 따라 어떻게 변화하고 있는지 살펴본다.

그림 4는, 실험별로 종합 효율값의 변화를 나타낸 그래프이다. 접근 속도는 차이가 있지만, 종합 효율값이 목표 효율값과 비슷하게 유지하려는 경향이 있음을 알 수 있다.

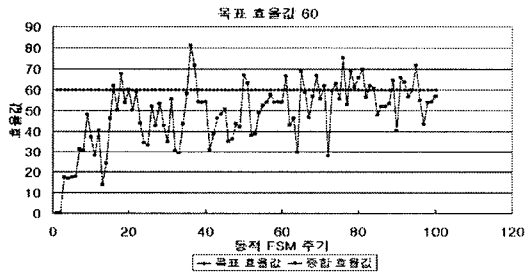
다음, Skill_use_log_npc, Skill_use_log_player 파일의 정보를 이용하여 각 실험별 게임 플레이어와 NPC의 행동 패턴 변화를 살펴본다. 목표 효율값이 100일 때 NPC와 게임 플레이어의 행동 패턴 변화를 나타낸 것이 그림 5이다. 게임 플레이어와 NPC의 행



(a) 목표 효율값이 100인 경우

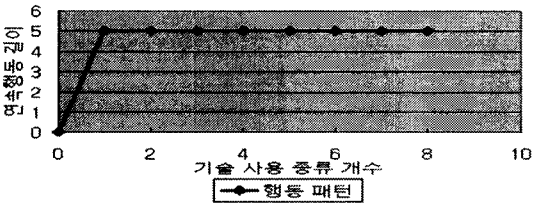


(b) 목표 효율값이 80인 경우

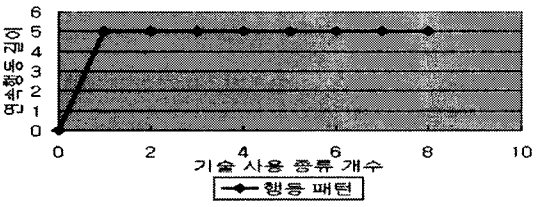


(c) 목표 효율값이 60인 경우

그림 4. 실험 1의 종합 효율값 변화 그래프



(a) 목표 효율값 100일 때 플레이어의 행동 패턴 변화



(b) 목표 효율값 100일때 NPC의 행동 패턴 변화

그림 5. 목표 효율값 100일 때 행동 패턴 변화 그래프

동 패턴이 같은 형태로 변화함을 확인할 수 있다. 목표 효율값 80, 60인 나머지의 경우도 거의 같은 형태로 나타났다.

4.4.2 실험 2 분석

실험 2 분석도 FSM_Log_TotalEfficiency 파일을 분석하여 종합 효율값의 변화를 분석하고, Skill_use_log_npc, Skill_use_log_player 파일을 분석하여 각 캐릭터의 행동 패턴을 살펴본다.

그림 6이 실험 2의 종합 효율값 변화 그래프이다. 게임 플레이어가 숙련자에서 초보자로 교체된 시점은 주기 100 이후이다. NPC의 종합 효율값은 그 전에 비해 100에서 20까지 매우 높아지거나 매우 낮아지는 경향을 보이다가 점차 다시 목표 효율값 60에 안정되는 모습을 볼 수 있다. 종합 효율값이 매우 높아진 것은, 숙련자에게 적응한 NPC가 초보자 게임 플레이어를 상대로 대결할 경우 초보자에게는 강한 NPC가 되었기 때문이다. 반대로 종합 효율값이 매우 낮아진 것은, 초보자 게임 플레이어를 상대로 NPC가 다시 적응하기 시작하여 매우 낮은 효율값을 가진 행동들을 실행하였기 때문이다.

결과적으로 동적 FSM이 동작할 때 숙련자에게 적응한 상태와 초보자에게 적응하고 있는 상태가 공존하여 그림 6과 같은 그래프의 형태가 나타났다. 주기 140 이후(초보자 게임 플레이어 기준 40 주기)에는 초보자 게임 플레이어에게 적응이 안정화되기 시작하여 숙련자에게 적응했을 때와 비슷한 형태의 그래프를 볼 수 있다.

그림 7은 실험 2에서 각 캐릭터 행동 패턴 변화를 나타내는 그래프이다. 각 캐릭터의 패턴이 급격하게 바뀌는 부분이 게임 플레이어가 숙련자에서 초보자로 바뀐 시점이다. 그래프 상에서 그 시점이 다르게

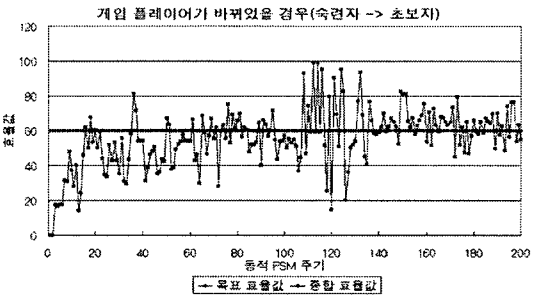


그림 6. 실험 2의 종합 효율값 변화 그래프

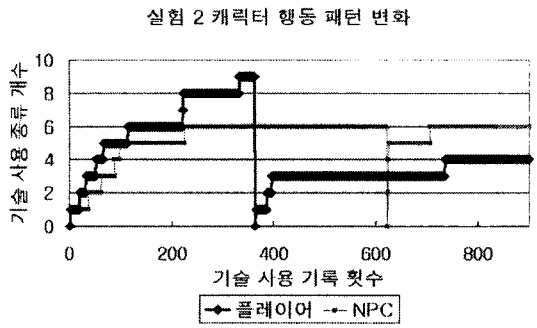


그림 7. 실험 2의 캐릭터 행동 패턴 변화 그래프

보이는 것은, 각 캐릭터가 기술을 사용할 때마다 기록하므로 게임 플레이어와 NPC는 서로 독립적인 행동으로 인해 그래프 상에서 다른 위치에 표현되기 때문이다. 로그 파일 분석 결과 NPC는 623번째 기록, 게임 플레이어는 364번째 기록부터 게임 플레이어가 숙련자에서 초보자로 바뀌었다. 게임 플레이어의 패턴을 살펴보면, 숙련자일 경우 기술 사용 개수가 초보자보다 많다. NPC의 경우에는 숙련자에게 적응되었던 NPC이므로 사용 기술 개수가 이전과 같지만, 그 이상은 증가하지 않는 것을 확인할 수 있다.

5. 결 론

본 논문에서는, 게임 플레이어에 맞추어 다양하게 행동할 수 있는 실시간 적응형 NPC를 생성하는 기법을 제안하였다. 이 기법은 다양한 행동 정보를 저장하고 있는 행동 DB를 이용하여 동적 FSM을 실시간으로 구성하고, 동적 FSM의 종합 효율값이 목표 효율값에 맞추어 가도록 FSM의 상태를 갱신함으로써 NPC가 적절한 행동을 실행할 수 있게 제어하는 것이다. 일반적인 정적 FSM을 사용하면 실시간으로 NPC의 행동을 다양하게 변화시킬 수 없기 때문에, 게임 실행 중 FSM이 실행되는 동안 FSM의 상태를 갱신할 수 있는 동적 FSM을 설계하여 사용하였다.

동적 FSM을 구성할 때에는 현재 동적 FSM의 종합 효율값을 목표 효율값으로 접근시키기 위해 동적 FSM 상태 갱신 알고리즘을 고안하여 적용하였다. 이 알고리즘은 간단한 원리로 동작하여 프로세서 자원을 많이 소모하지 않는 특징을 가지고 있다.

이 기법을 실험하기 위해, 실험용 게임을 제작하고 세가지 목표 효율값을 설정하고 FSM의 종합 효율값이 접근해가는 형태를 분석하였으며, 숙련자와 초보자 플레이어로 하여금 종합 효율값의 변화를 분석하였다. 어느 경우에도 종합 효율값이 목표 효율값으로 접근해가면서 NPC가 플레이어에게 적응해가는 것을 알 수 있었다. 본 연구에서 제안한 동적 FSM을 이용한 적응 기법은 FSM을 이용하여 NPC를 제어하는 모든 게임에 적용될 수 있는 특징이 있다.

참 고 문 헌

[1] Penny Baillie-de Byl, *Programming Believable Characters for Computer Games*, Charles River Media, Massachusetts, 2004.

[2] A. Ram, S. Onta~non, and M. Mehta, "Artificial Intelligence for Adaptive Computer Games," *FLAIRS-07*, pp. 22-29, 2007.

[3] 표준 국어 대사전, 국립국어원, 1999.

[4] G. Andrade, G. Ramalho, A. Gomes, and V. Corruble, "Dynamic Game Balancing : an Evaluation of User Satisfaction," *AIIDE* 2006, pp. 3-8, 2006.

[5] G. Andrade, G. Ramalho, H. Santana, and V. Corruble, "Challenge-Sensitive Action Selection: an Application to Game Balancing," *IAT-05*, pp. 194-200, 2005.

[6] 조병헌, "지능형 게임 캐릭터를 위한 학습 및 적응 방법에 관한 연구," 국민대학교 대학원 전자공학과 박사 학위 논문, 2004.

[7] P. Spronck, *Adaptive Game AI*, SIKS Dissertation Series 2005-06, 2005.

[8] J. Dinerstein and P.K. Egbert, "Fast multi-level adaptation for interactive autonomous characters," *ACM TOG*, Vol.24, pp. 262-288, 2005.

[9] 엄상원, "인공지능 기법을 이용한 사용자 상호작용 게임 난이도 조절 알고리즘," 중앙대학교 첨단대학원 석사 학위 논문, 2003.

양 정 모

2006년 2월 동국대학교, 컴퓨터공학과 (공학사)
2008년 2월 동국대학교, 컴퓨터공학과 대학원 (공학석사)
관심분야 : 게임 인공지능, 컴퓨터 네트워크

조 경 은

1993년 2월 동국대학교, 전자계산학과(공학사)
1995년 2월 동국대학교, 컴퓨터공학과 대학원(공학석사)
2001년 8월 동국대학교, 컴퓨터공학과 대학원(공학박사)
2003년 9월~2005년 8월 동국대학교 정보산업대학 컴퓨터멀티미디어공학과 전임강사
2005년 9월~2007년 6월 동국대학교 정보산업대학 컴퓨터멀티미디어공학과 조교수
2007년 7월~현재 동국대학교 영상미디어대학 게임멀티미디어공학과 조교수
관심분야 : 컴퓨터 게임 알고리즘, 게임 인공지능, 멀티미디어 정보처리

엄 기 현

1975년 2월 서울대학교 공과대학 응용수학과 공학사
1977년 2월 한국과학기술원 전산학과 이학석사
1994년 2월 서울대학교 대학원 컴퓨터공학과 공학박사
1978년 3월~2007년 6월 동국대학교 컴퓨터멀티미디어공학과 정교수
2007년 7월~현재 동국대학교 영상미디어대학 게임멀티미디어공학과 교수
1995년 3월~1999년 2월 동국대학교 정보관리처장 역임
2001년 3월~2003년 2월 동국대학교 정보산업대학 학장 역임
2005년 3월~현재 한국 게임학회 자문위원
1998년 12월~2001년 12월 한국 멀티미디어학회 부회장, 자문위원, 수석부회장 역임
2007년 1월~2007년 12월 한국멀티미디어학회 회장
관심분야 : 게임시스템 및 구조 설계, 멀티미디어응용시스템, 멀티미디어데이터베이스