한국컴퓨터정보학회 동계학술대회 논문집 제27권 제1호 (2019. 1)

유한 상태 기계를 이용한 몬스터 AI 구현에 관한 연구

조재원O, 방정원*
O*청강문화산업대학교 게임콘텐츠스쿨
e-mail: icew03@gmail.comO, jwbang@ck.ac.kr*

A study on The Implementation of Monster AI using Finite-State Machine

Jae-Won JoO, Jung-Won Bang*
O*School of Game, Chungkang College of Cultural Industries

● 요 약 ●

게임에서 장르를 불문하고 모든 몬스터와 NPC는 AI를 가지고 있다. 따라서 적 몬스터 캐릭터와 전투를 즐기는 액션 게임의 경우 그만큼 인공지능이 게임 안에서 차지하는 비율이 높다고 할 수 있을 것이다. 본 논문에서는 FSM, HFSM, BT와 같은 AI 기법을 비교하여 분석하였다. 각 기법에는 주의해야 할 점이 명확하게 존재하기 때문에 구체적으로 어떠한 문제점들이 존재하는지에 대한 결과를 얻는데 연구 목적이 있다. 따라서 몬스터 AI를 구현할 때 각 인공지능 기법의 장단점을 고려하여 설계하여 유지 보수를 줄이는 방법을 연구해야 한다는 것을 확인할 수 있었다.

키워드: FSM(Finite-State Machine), HFSM(Heirarchical-Finite-State Machine), BT(Behavior Tree)

I. Introduction

게임에서 장르를 불문하고 모든 몬스터와 NPC는 AI를 가지고 있다. 따라서 적 몬스터 캐릭터와 전투를 즐기는 액션 게임의 경우 그만큼 인공지능이 게임 안에서 차지하는 비율이 높다고 할 수 있을 것이다.

본 논문에서는 이러한 몬스터 AI를 기획자 또는 협업하는 프로그래머가 쉽게 확인하고 설계할 수 있도록 도표로써 나타내기 위해 FSM을 이용하여 몬스터 AI를 구현하였다.

II. Preliminaries

AI (Artificial Intelligence)란, 인공지능으로 인간의 학습능력, 추론 능력, 지각 능력, 이해 능력 등을 컴퓨터 프로그램으로 실현한 것이다.

유한 상태 기계 (Finite-State Machine, FSM)는 AI 기법 중 하나로 유한한 개수의 상태를 가지는 추상 기계이다. 한 번에 보통 하나의 상태만을 가지며 FSM은 어떤 이벤트에 의해 한 상태에서 다른 상태로 변화할 수 있으며 이를 전이라고 한다.[1]

FSM은 몬스터 AI를 제작할 때 가장 전통적으로 사용되는 방법이며 쉬운 개념의 정립, 각 상태와 형태가 코드 상이 아닌 도표로써 나타내짐으로써 AI의 개념을 프로그래머가 아닌 기획자 또는 제3자가 쉽게 확인, 설계가 가능하다는 장점이 있다.

따라서 본 논문에서는 위 장점들을 활용하여 몬스터들의 AI를 구현하였다.

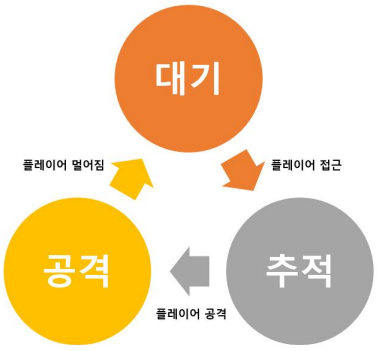

Fig. 1. FSM

III. The Proposed Scheme

위의 방식을 사용하여 기획서에 맞게 여러 가지 형태의 몬스터 AI를 구현할 수 있다. 그러나 여기에는 고려해야 하는 문제가 있다. 게임 개발 도중 기획서는 언제나 수정될 가능성이 있으며, 몬스터의 구조 또한 더 복잡하게 변경될 수도 있기 때문이다.

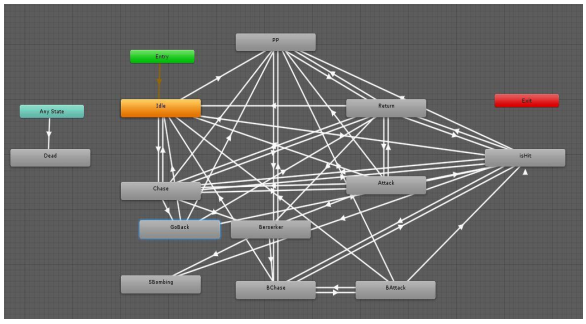

Fig. 2. 복잡해진 FSM 구조

따라서 복잡해진 몬스터의 AI 구조를 FSM만으로 처리하는 데는 한계가 있다. 그 해결 방법으로는 두 가지를 생각해 볼 수 있다.

첫째는, HFSM을 사용하여 FSM 들을 그룹화하고 계층화시키는 방법이다.

HFSM(Heirarchical-Finite-State Machine)란 최근까지 게임 AI에서 가장 많이 사용되는 방식으로, FSM 들을 그룹화하고 계층화함으로써 특정 문맥을 가진 상태를 재사용할 수 있게 해준다. FSM과 마찬가지로 현재의 상태와 형태를 파악하기 쉬운 장점이 있다. 주의해야 할 점은 HFSM 이 어느 정도의 확장성을 가지기는 하지만, (하위) 상태를 모듈화하는 기능을 가지고 있지는 않다는 점이다. 특정 상태에서 전이되어야 한다는 조건

이 붙기 때문에 같은 상태를 다른 문맥에서는 사용할 수 없는 것이다.[2]

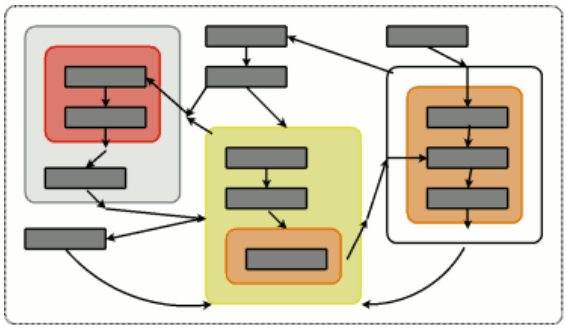

Fig. 3. HFSM

Behavior Tree를 사용하여 로직을 캡슐화하면, 이 문제를 쉽게 해결할 수 있다.

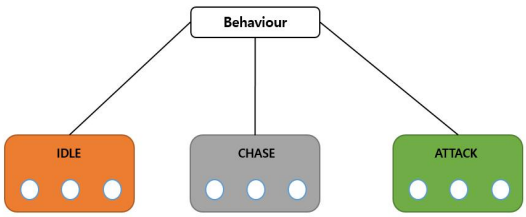

Fig. 4. Behavior Tree

Behavior Tree란 HFSM과 달리 다른 목적이나 상황에 따라 상태를 재사용할 수 있도록 하기 위한 행동트리라 할 수 있다. 로직을 캡슐화함으로써 상태의 모듈성을 증가시키는데, 일반적으로 BT를 구현하기 위해서는 스택을 사용하게 된다.[3] 위 방식을 적용하면 모듈 재활용이 용이하고 기존 HFSM보다 유지 보수가 간편하기 때문에 개발 중간 기획서의 복잡한 AI를 안정적으로 구현할 수 있었다. 본 논문에서는 몬스터의 AI 구조가 뒤늦게 변경되었기 때문에 FSM을 최종적으로 선택하여 몬스터들의 AI를 구현하였다.

IV. Conclusions

AI 기법은 나날이 발전하고 있으며, 새로운 기술들이 나타나고 있다. 몬스터 AI의 구현에는 FSM, HFSM, Behavior Tree 같은 다양한 방법이 있다. 몬스터 AI를 구현할 때는 각 인공지능 기법의 장단점을 고려하여 설계하면서 유지 보수를 줄일 수 있는 방법에 대한 연구가 필요할 것으로 예상된다.

REFERENCES

[1] http://showmiso.tistory.com/156
[2] http://aigamedev.com/open/article/hfsm-gist/
[3] http://lifeisforu.tistory.com/327