Journal of Korea Multimedia Society Vol. 11, No. 1, January 2008(pp. 76-87)

단계적 퍼지 시스템 전략모듈을 지원하는 에이전트기반 게임 플랫폼

이원희[†], 김원섭[††], 김태용[†††]

요 약

하드웨어의 성능이 높아질수록 게임 유저들은 높은 수준의 컴퓨터 그래픽, 편리한 유저 인터페이스, 빠른 속도를 가진 네트워크 그리고 영리한 게임 인공지능을 요구하고 있다. 하지만 현재 게임 인공지능 개발은 개발자 혼자 하거나 한 회사의 개발팀에서만 이루어질 뿐이다. 그래서 자신이 혹은 회사에서 개발한 게임 인공지능의 성능이 어느 정도인지 검증을 하기 힘들고 높은 수준의 게임 인공지능을 개발하기 위해 필요한 기본 게임 인공지능기술들이 부족하다. 본 논문에서는 기존의 게임인공지능 플랫폼들의 장, 단점을 알아보고 게임인공지능 플랫폼의 설계 시 고려해야 할 점을 고찰한다. 이것을 바탕으로 전략적 위치를 찾아주는 모듈이 있어 개발자 들이 손쉽게 게임 인공지능을 구현 하고 인공지능 테스트가 가능한 에이전트기반 게임 플랫폼인 Darwin을 제안한다. 그리고 Darwin에서 제공하는 전략적 모듈을 사용하여 제작한 에이전트를 만들어 수행결과를 평가한다.

Agent-Based Game Platform with Cascade-Fuzzy System Strategy Module

WonHee Lee[†], WonSeop Kim[††], TaeYong Kim[†††]

ABSTRACT

As hardware performance rises, game users demand higher computer graphic, more convenient UI(User Interface), faster network, and smarter AI(Artificial Intelligence). At this time, however, AI development is accomplished by a co-development team or only one developer. For that reason, it's hard to verify that AI performance and basic game AI technology is lacking for developing high-level AI. Searching the merits and demerits of existing game AI platforms, we investigate main points to consider when designing game AI platforms in this paper. From this we suggest Darwin, a game platform, based on agent that developers embody AI easily and capable of proposing AI test with module that makes them find strategic position. And then evaluate achievement results through making agent used strategic module that Darwin offers.

Key words: Intelligent Agent(지능 에이전트), Fuzzy Control(퍼지 제어), Game Strategy(게임 전략)

1. 서 론

디지털 게임은 탄생한지 50 년이 채 안됨에도 불구하고 엔터테인먼트 분야에서 상업적으로 가장 빠른 속도로 발전해 왔다. 상업용 게임의 성공은 관련된 하드웨어와 소프트웨어의 발전에 기인하고 있으

※ 교신저자(Corresponding Author) : 김태용, 주소 : 서울특별시 동작구 흑석동 아트센터 지하1층 106호(608-743), 전화 : 02)812-5717, FAX : 02)814-5404, E-mail : kakybear@gmail.com
접수일 : 2007년 6월 19일, 완료일 : 2007년 11월 12일
[†] 준회원, 중앙대학교 첨단영상대학원 (E-mail : kakybear@dreamwiz.com)
[††] 준회원, 중앙대학교 첨단영상대학원 (E-mail : refarde@gmail.com)
[†††] 정회원, 중앙대학교 첨단영상대학원
※ 본 연구는 ITRC(Information Technology Research Center)와 서울시 산학연 협력사업의 지원으로 수행되었음.

며 하드웨어의 성능이 높아질수록 게임 유저들은 높은 수준의 컴퓨터 그래픽, 편리한 유저 인터페이스, 빠른 속도를 가진 네트워크 그리고 영리한 게임 인공지능을 요구한다. 특히 게임 인공지능의 경우에는 화려한 영상과 빠른 속도가 우선시 되는 게임 산업의 특수성으로 인하여 과거 게임제작비용과 하드웨어의 성능에 제한적으로 발전해 왔다. 게임과 관련한 하드웨어의 발달은 게임 디자이너가 지속적으로 다른 게임과의 차별성을 줄 수 있는 참신한 방법을 찾게 하였고 많은 게임 회사들이 이러한 방법으로 인공지능을 선택하였다. 'Sims'와 'black and white' 시리즈는 이러한 특성을 가진 대표적인 성공한 상업용 게임이다[1,2].

이와 같이 상업용 게임 인공지능의 발전과 더불어 학계에서도 게임 인공지능의 개발과 환경에 관한 많은 연구가 진행되어 왔다. Robocode는 IBM이 자바의 보급과 교육을 목적으로 자바를 응용하여 설계된 게임 플랫폼이다[3]. Robocode는 에이전트기반 게임 플랫폼이지만 게임 환경이 종속적이어서 다양한 장르의 게임의 인공지능을 구현하기가 쉽지 않다. 그리고 네트워크로 연결되어 있지 않아 구현한 인공지능의 성능테스트에 어려움이 있다. Gamebots는 University of Southern California's Information Sciences Institute와 Carnegie Mellon University에 의해 공동 개발 진행된 프로젝트로서 여러 개의 Bot을 위한 멀티 에이전트 개념을 포함하였다[4,5]. 하지만 Gamebots의 경우 멀티 에이전트를 지원하는 플랫폼으로 네트워크로 연결되어 인공지능의 성능 테스트가 자유롭게 이루어지는 반면 언리얼 토너먼트라는 상용게임플랫폼을 사용함으로써 개발자들이 개발하기 힘든 환경이 주어진다.

본 논문에서는 이러한 기존 게임인공지능 플랫폼이 어떻게 설계되었는지 알아보고, 기존 게임인공지능 플랫폼들의 장단점을 알아본다. 이것을 바탕으로 퍼지 로직을 사용하여 작성된 전략 모듈의 사용으로 쉽게 인공지능 개발이 가능하고 인공지능의 성능을 테스트하기 쉬운 에이전트기반 게임 플랫폼을 제안한다. 그리고 아케이드스타일의 게임을 구현하여 전략 모듈을 적용한 에이전트를 작성한 후 전략 모듈이 적용 안 된 에이전트와 평가하고자 한다. 논문의 구성은 다음과 같다. 2장은 기존 게임인공지능 플랫폼에 대한 장단점을 분석한다. 3장은 기존 게임인공지능 플랫폼의 단점을 보완한 새로운 게임인공지능 플랫폼 다윈(Darwin)의 구조를 제안한다. 4장은 제안된 게임인공지능 플랫폼(Darwin)의 전략 모듈을 사용하여 작성한 에이전트와 일반 에이전트를 실험하여 실험 결과를 설명한다. 마지막으로 5장은 결론과 추후 연구 과제에 대하여 기술한다.

2. 기존 인공지능 플랫폼

게임에서 게임인공지능의 성능을 높이기 위해 일반적으로 사용하는 기법들은 다음과 같다[4]. 게임인공지능 플랫폼은 게임 환경에서 상황 변화를 게임인공지능 플랫폼이 인지하고 반응할 수 있어야 하며 게임인공지능 플랫폼이 게임성능을 저하시켜서는 안 된다. 이 때문에 게임인공지능 플랫폼은 실시간성을 지원해야한다. 또한 게임인공지능 플랫폼은 게임 환경에 독립적으로 작동하고 게임 환경에서 인지하는 정보를 이용해야 하고, 게임인공지능 플랫폼에서 수정한 정보를 게임 환경에 적용 가능해야 하는 독립성과 범용성을 가지고 있어야 한다. 그리고 인공지능 디자이너가 요구한 사항을 개발자가 쉽게 구현할 수 있어야 하는 쉬운 개발 환경을 필요로 한다. 마지막으로 각각의 부분을 모듈화 하여 통해 새로운 인공지능의 추가/수정이 용이해야 하는 모듈화가 충족되어야 한다. 2장에서는 위에 제시된 기준으로 이전에 연구, 개발된 게임인공지능 개발 플랫폼들의 구조와 특징을 알아본다.

2.1 Robocode

Robocode는 자바 플랫폼을 응용하여 설계되었다. Robocode는 그림 1과 같이 대부분의 현대 자바 VM(Virtual Machine)이 제공하는 비 선점 형 스레딩을 응용하였다. 시뮬레이션 되고 있는 각 로봇이 자체적인 자바 스레드 상에 있다. 또한, Robocode는 공유된 자원에서 발생할 수 있는 잠재적인 문제들을 완화하기 위해 전투 관리자 스레드와 로봇 스레드 간에 매우 느슨한 결합을 사용한다. 이러한 느슨한 결합을 구현하기 위해 각 로봇 스레드에 자체적인 이벤트 큐를 주었다. 그러면 로봇의 이벤트들이 로봇의 자체 스레드에 보내져 처리된다. 이러한 스레드당 큐잉은 전투 관리자 스레드와 로봇 스레드 사이에, 그리고 로봇 스레드들 사이에 발생할 수 있는 충돌을

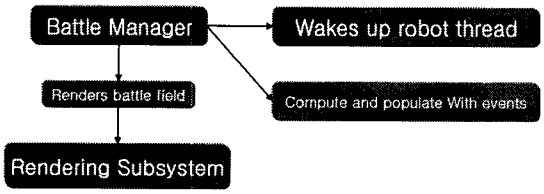

그림 1. Robocode

없애준다. 하지만 자체적인 게임 환경을 가짐으로써 게임 환경에 독립적이지 못하여 여러 가지 장르의 게임인공지능을 구현 및 테스트하기 힘들고 자바교육을 목적으로 하고 있기 때문에 인공지능환경을 지원해 주지 않아 개발이 쉽지 않다.

2.2 Gamebots

Gamebots는 University of Southern California's Information Sciences Institute와 Carnegie Mellon University에 의해 공동 개발 진행된 프로젝트이다. 이 프로젝트는 인터넷을 통한 멀티 플레이어를 지원하는 언리얼 토너먼트 게임플랫폼을 기반으로 한 인공지능 개발 지원 멀티 에이전트(multi-agent)시스템이다. 그림 2에서 보는 것과 같이 Gamebots는 네트워크 소켓을 통해 보트(Bot) 클라이언트에 연결되어 캐릭터를 조정할 수 있게 해주는 UT 모듈에 있다. Gamebots서버는 네트워크를 통해 캐릭터의 인지 정보를 제공한다. 이때 에이전트가 인지 할 수 있는 정보는 인간 플레이어가 인지하는 범위를 벗어나지 않는다. 제공 되어진 정보에 기반 하여 클라이언트(Bot 또는 인간 플레이어)는 캐릭터가 어떤 행동을 할 것인지 결정할 수 있고, 네트워크를 통해 이동하거나

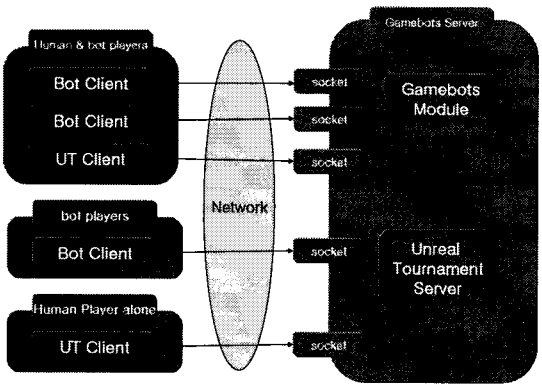

그림 2. Gamebots

전투하기, 대화하는 등의 명령을 전달 할 수 있다. 하지만 게임 환경으로 언리얼 토너먼트라는 상용게임플랫폼을 사용함으로써 언리얼 토너먼트플랫폼을 구매하지 않은 많은 개발자들이 개발해 볼 수 없는 환경이 주어진다.

3. 에이전트기반 게임플랫폼 : Darwin

2장에서 알아본 기존 게임인공지능 플랫폼의 단점을 만족시키고 플랫폼 설계 시 고려해야할 점을 기반으로 하여 본 논문에서는 게임인공지능의 성능테스트가 용이하고 전략 모듈을 지원하여 개발을 쉽게 해주는 게임인공지능 플랫폼인 Darwin을 제안한다.

3.1 전체구조

Darwin은 그림 3과 같이 AI-레벨, 게임-레벨 2가지 영역으로 구분되어 있다.

게임 개발자는 Darwin에서 제공하는 템플릿을 사용하여 DLL 또는 Lua를 제작한다. 게임 인공지능 개발자가 DLL 또는 Lua를 제작할 때 Game-레벨의 Darwin-매니저는 신경망, 유전자 알고리즘, 상태머신 등의 인공지능라이브러리를 제공하여 개발의 효율성을 향상 시켜준다. 게임인공지능 개발자가 개발한 DLL, Lua는 Darwin-매니저의 Darwin-로더에 의해서 로드되고 게임에 등장하는 게임 에이전트에게 지능을 부여한다. 지능이 부여된 게임 에이전트의 동작은 Darwin-뷰어를 통해서 확인이 가능 하다.

그림 4는 주요역할을 처리하는 Darwin-매니저의 클래스 구조도이다. Darwin-메인에서 Darwin-로더를 처리하는 RunLoader()를 수행하여 Darwin-러너에게 정보를 보내주는 역할을 Darwin-메인의 Update()에서 수행을 한다. Darwin-러너는 기본적인 Bot의 구조를 상속받은 AgentBot클래스를 이용

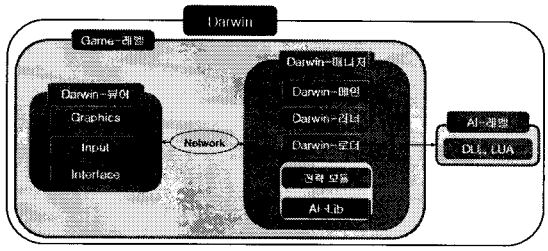

그림 3. Darwin

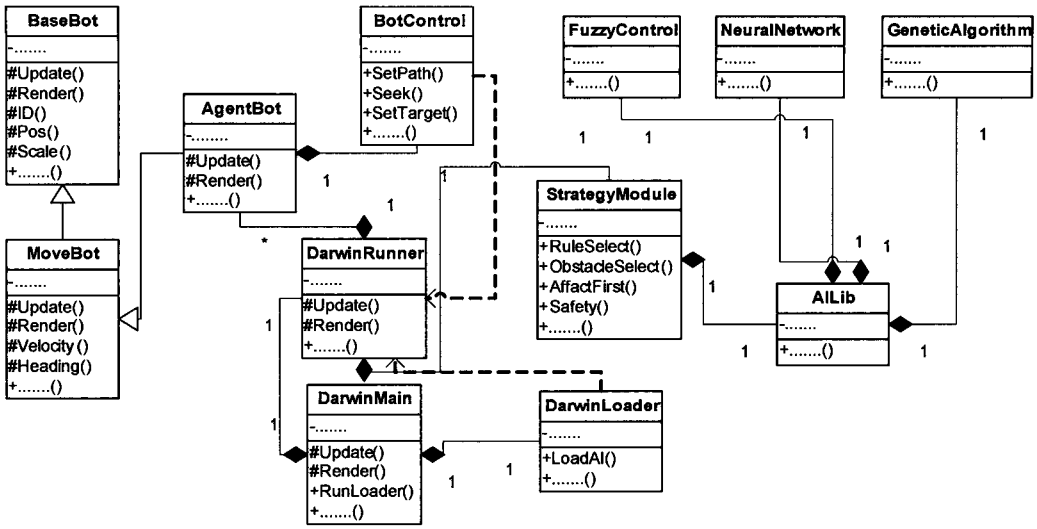

그림 4. Darwin-매니저

하여 게임에 필요한 인공지능개체를 생성해낸다. AgentBot은 생성 될 때 Darwin-로더에서 받은 에이전트의 정보(룰 등)를 이용하여 생성이 되고 그것을 기본으로 전략모듈 에이전트의 움직임 룰 값을 정할 때 이 정보를 사용한다. 그리고 기본적인 움직임 이외에 고급제어를 할 수 있는 BotControl클래스를 지니고 있어 A*알고리즘을 이용한 길 찾기 등의 상위 제어를 사용할 수 있다. Darwin-러너는 생성된 에이전트의 정보를 읽어가며 에이전트에 해당하는 룰대로 현재 사용하는 맵의 정보와 함께 에이전트의 정보를 StrategyModule클래스에 정보를 넘겨 해당하는 값을 도출한다.

3.2 각 부분별 기능

3.2.1 Darwin-뷰어

Game-레벨의 구조를 이루는 부분으로 게임 환경으로써 게임의 화면을 보여주고 사용자의 입력제어를 담당하는 부분이다. 인공지능처리부와의 독립된 환경을 위해 인공지능처리부와는 네트워크로 연결되어 있다. 때문에 Darwin-뷰어는 게임화면과 게임을 플레이하는 유저의 입력정보만을 처리하면 되기 때문에 그래픽부와 입력, 인터페이스부로 나누어져 있다. 처리하는 역할은 표 1과 같다.

3.2.2 Darwin-매니저

Darwin-뷰어와 마찬가지로 Game-레벨의 구조를 이루는 부분으로 인공지능처리부로써 게임 에이전트를 총괄하는 곳이다. 작성된 인공지능 DLL이나 Lua 파일을 읽어 들여 게임 환경과 통신을 통해 게임인공지능을 처리하며 Darwin을 관리한다. 인공지능처리부는 게임 환경과 독립적 이어야 하기 때문에 게임 환경과 통신을 담당하는 곳이 있어야 하며 인공지능의 모듈화를 통해 인공지능의 추가 수정이 용이해야 함으로 인공지능파일을 관리하는 곳이 있어야한다. 그리고 게임인공지능의 쉬운 개발을 위해 인공지능라이브러리를 제공하는 곳이 있어야한다. 때문에 Darwin에서는 Darwin-메인, Darwin-러너, Darwin-로더, AI-Lib, 전략 모듈과 같이 5가지부로 나누어 놓았다. 각각에서 처리하는 역할은 표 2와 같다.

표 1. Darwin-뷰어

그래픽	에이전트와 지형들을 랜더링하여 스크린에 보여준다.
입력	카메라 제어 같은 입력제어를 담당한다.
인터페이스	인공지능 처리부인 Darwin-매니저와의 통신으로 각각의 에이전트에 정보를 갱신하며 통신은 TCP/IP프로토콜을 사용한다.

3.2.3 AI-레벨

AI-레벨은 Darwin에서 제공하는 템플릿을 처리하는 곳이다. Darwin에서 제공하는 템 플릿은 인공지능의 추가/수정이 용이하도록 Lua-Script와 DLL로 작성되어 제공된다. DLL이나 Lua로 모듈화 되어

표 2. Darwin-매니저

Darwin-메인	Darwin-러너와 Darwin-로더의 제어를 관장하며 Darwin-뷰어의 Interface와 정보를 통신하여 에이전트정보를 갱신한다.
Darwin-러너	Darwin-로더에서 읽어 들인 인공지능 정보를 실제적으로 처리하는 부분이다.
Darwin-로더	인공지능 개발자가 개발한 인공지능을 로드하고 그 정보를 게임 에이전트에게 부여한다.
AI-Lib	Neural, Genetic Algorithm, FSM같은 Low 레벨의 인공지능라이브러리를 제공한다.
전략 모듈	High 레벨의 인공지능라이브러리로써 상황에 따른 전략적 위치를 계산하는 고수준의 인공지능을 제공한다. Darwin의 전략 모듈에서는 중요위치 선점을 위한 단계적 퍼지 로직을 사용한 모듈을 제공함으로써 게임 내에서 전략적 위치를 선점하여 게임을 유리하게 이끌어 가게 한다.

진 게임 인공지능은 Game-레벨의 Darwin-로더에서 처리되어진다. 그리고 Darwin-매니저의 AI-Lib와 전략모듈을 사용하여 인공지능의 구현을 용이하게 한다.

4. 중요위치 선점을 위한 Cascade-Fuzzy Logic 전략 모듈

Darwin의 AI-Lib를 사용하여 High 레벨 Query를 처리하는 부분이다. 전략 모듈에서는 Cascade-Fuzzy Logic을 사용한 위치선정 시스템을 사용하여 에이전트가 게임에서 상황에 따라 이동할 위치를 자동으로 선정해주는 시스템을 제공하여 개발을 쉽게 해주어 개발시간을 절약하게 도와준다.

4.1 퍼지 로직

퍼지 논리는 불린(Boolean)논리의 상위 개념으로 포함의 정도를 표현하기 위해 제안 되었다. 그림 5의 구조를 가지는 퍼지 논리시스템은 일반적으로 세 가지 부분으로 구성된다. 세 가지 부분은 하나의 명확한 값(crisp value)으로 측정된 입력 변수의 값을 적절한 퍼지 값으로 바꾸는 퍼지 화(fuzzification), 조건부와 결론부의 조건문으로 이루어지는 퍼지추론 규칙(fuzzy inference urles), 출력부 전체집합에서 정의된 퍼지 값을 명확한 비 퍼지 값으로 변환시켜주는 작업이 비 퍼지 화(defuzzification)이다.

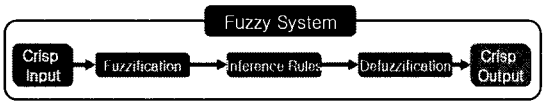

그림 5. 퍼지 논리 시스템 구조

퍼지 화는 명확한 값에서 퍼지 값을 생성하는 과정으로 퍼지 집합에 의해 수행되며, 소속 함수(membership function)를 사용한다. 그러므로, 퍼지 화의 성능은 소속 함수의 효율성에 의존적이다. 예를 들어, 간단한 소속 함수는 빠르게 퍼지 화를 수행할 수 있는 반면, 복잡한 소속 함수는 빠르게 퍼지 화를 진행시킬 수 없게 된다.

추론 규칙은 일반적으로 규칙 기반 시스템처럼 "if-then"형식으로 표현되며, 전체 규칙은 여러 개의 복수 입력, 복수 출력의 퍼지 조건문들로 구성되는 것이 보통이다. 하지만, 퍼지 추론 규칙은 명확한 값만을 사용하고 처리하는 규칙 기반시스템과는 다른 특성을 가진다. 즉, 퍼지 논리에서는 working memory에 사용되는 심볼 또한 퍼지 값으로 표현되며, 규칙을 추론하는 추론기 또한 조건문이 참을 이루는 정도에 따라 결론을 추론하게 된다. 일반적으로, 퍼지규칙을 추론하는 방법으로는 Mamdani가 제안한 min-max method를 사용한다.

비 퍼지 화는 퍼지 화와는 반대로 퍼지 값에서 명확한 값으로의 전환을 목적으로 한다. 비 퍼지 화를 하는 주요 방법으로는 최대값 방법, 최대평균법, 무게중심법이 있으며, 그 중에서 무게중심법이 가장 많이 사용된다. 무게중심 법(center of gravity method)은 합성된 출력 퍼지집합의 무게중심을 구하여 그 해당하는 값을 입력으로 사용하는 방법이다[6-8].

4.2 Cascade-Fuzzy Logic을 사용한 위치 선정

게임에서는 상대와 대결을 할 때 상황에 따라 에이전트가 이동해야할 위치가 다르게 된다. 그리고 그 상황이라는 것은 논리적으로 항상 정확하게 설명을 할 수 없는 부분이 많다. 그래서 본 전략 모듈은 단계적으로 퍼지 로직을 이용하여 map의 cell에 대한 전략적 위치 값 을 얻어낸다. 전략적 위치는 우선 상대편의 공격을 피할 수 있는 장소가 있어야 하고 상대편의 공격을 회피 할 곳을 찾았다면 상대와 교전을 하여

이길 수 있어야 한다. 위의 요소를 고려하여 Darwin의 전략 모듈에서는 다음과 같은 요소를 사용하였다.

(1) Obstacle-Select - 엄폐물과의 관계

(2) Attack-First - 공격 할 수 있는 정도

(3) Safety - 위험도의 정도

그림 6은 전략 모듈의 전략적 위치를 찾는 개요도이다.

전략 모듈은 전략적 위치를 찾기 위해서 Map정보와 에이전트정보를 사용하게 된다. Map의 정보로는 cell의 위치정보와 cell이 높은-지형인지, 낮은-지형인지에 대한 입력 값을 받는다. 에이전트정보로는 에이전트의 에너지, 공격력, 방어력, 사정거리, 시야범위, 이동속도, 공격지연, 현재위치, 생존한 아군의 수, 생존한 적군의 수, 목표 에이전트와의 거리, 목표 에이전트를 기준으로 한 주변의 아군/적군 에이전트 수, 현재위치를 기준으로 한 주변의 아군/적군 에이전트 수, 시야내의 아군/적군의 숫자, 행동양식에 대한 입력 값을 받는다.

다음은 전략 모듈의 처리과정이다.

(1) 입력 값을 받은 후 Rule-Select과정을 거치게 된다. Rule-Select과정은 에이전트의 행동양식에 따라 Obstacle-Select, Attack-First, Safety의 퍼지 논리화과정에서 추론 규칙을 정할 때 결과 값을 행동양식에 맞게 선택하게 해주는 과정이다.

(2) 퍼지시스템인 Obstacle-Select, Attack-First, Safety에서는 입력 값을 각각의 모듈에 맞게 퍼지화 한다. 그리고 추론 규칙을 거치는 데 규칙 값으로서는 Rule-Select과정에서 선택된 결과 값을 사용하고 방식은 min-max method를 사용하였다. 그리고 추론 규칙과정을 거쳐 도출된 퍼지 값은 무게중심법을 사용하여 비 퍼지 화하였다.

(3) 전략 Position은 전략적 위치 값을 도출하는 과정이다. 퍼지시스템으로써 입력 값으로 Obstacle-Select, Attack-First, Safety에서 도출된 값을 입력 값으로 사용한다. 전략 Position 모듈도 추론 규칙은 min-max method를 사용하였고 무게중심 법을 사용하여 비퍼지화 하였다.

(4) 검사범위의 cell당 전략 모듈을 거치게 되고 최종적으로 가장 값이 높게나온 cell을 전략적 위치로 판명하게 된다.

다음은 각각의 Darwin의 전략 모듈에서 사용한 Fuzzy 모듈이다.

가. Obstacle-Select

- 적의 공격을 피할 수 있는 엄폐물을 찾아주는 모듈이다. 좋은 엄폐물을 찾기 위해서 엄폐물과 나와의 거리, 엄폐물 주위의 적의 숫자를 입력 값으로 하였고 각각의 소속 함수는 그림 7과 같

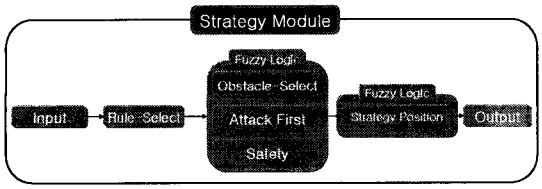

그림 6. 전략 모듈의 개요도

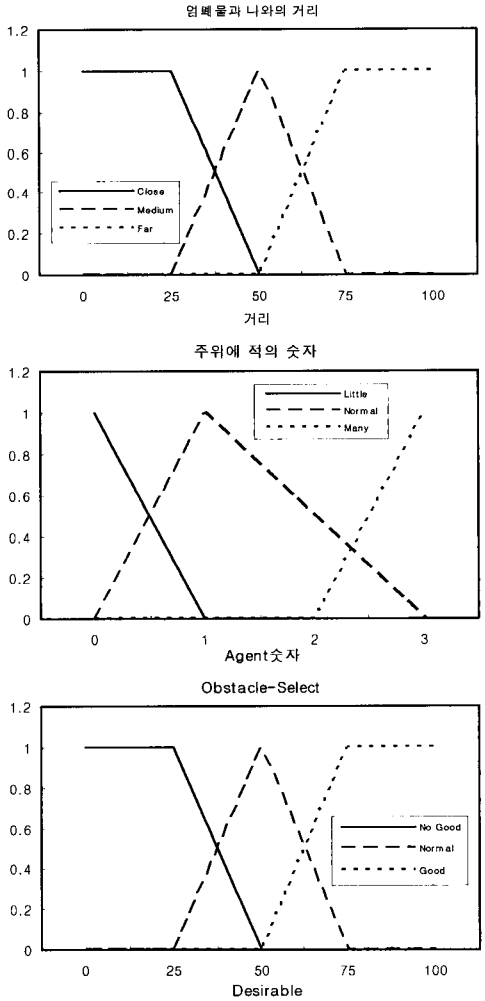

그림 7. Fuzzy Variable for Obstacle-Select

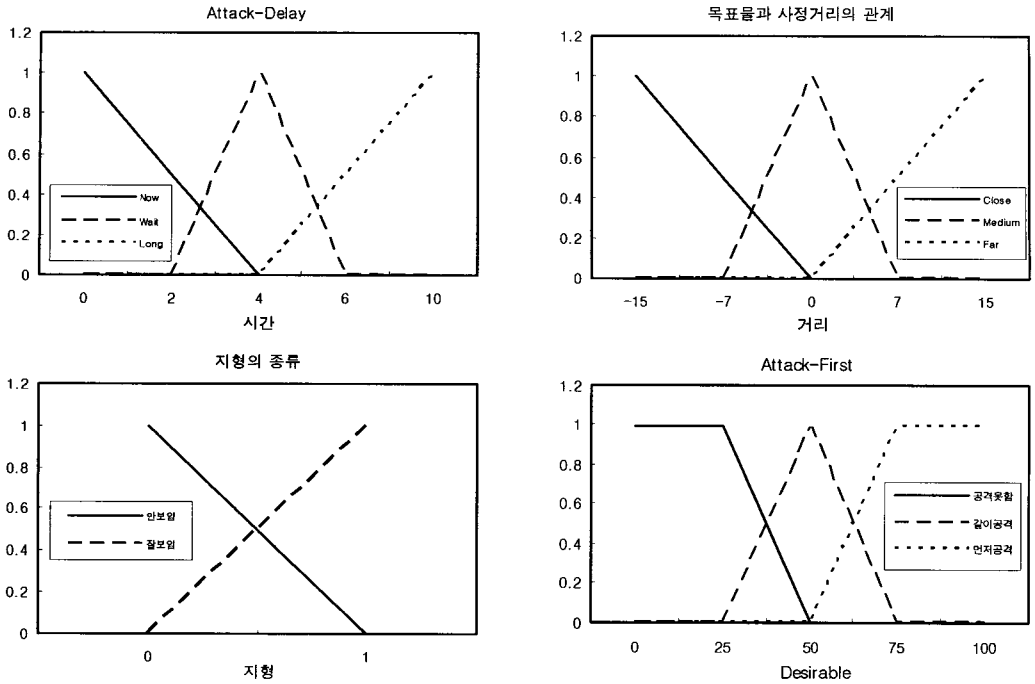

그림 8. Fuzzy Variable for Attack-First

다. 이 2개의 요소로부터 Obstacle-Select값을 도출한다. Obstacle-Select는 2개의 요소가 3개씩의 퍼지 값을 가지고 있고 추론규칙은 min-max method를 사용하였기 때문에 총 9개의 룰 셋을 가지고 있다.

나. Attack-First

- 현재 위치에서 공격을 할 수 있는가를 알아보는 것으로 아래와 같은 입력 값을 가진다. 현재 남은 공격지연, 목표 에이전트거리와 사정거리의 관계, 현재위치의 시야범위 관계를 입력 값으로 한다. 이 3개의 입력 값으로부터 Attack-First 값을 도출한다. 각각의 소속 함수들은 그림 8과 같다.

  Attack-First는 2개의 요소가 3개씩의 퍼지 값을 가지고 있고 1개의 요소가 2개의 퍼지 값을 가지고 있다. 추론규칙은 min-max method를 사용하였기 때문에 총 18개의 룰 셋을 가지고 있다.

다. Safety

- 자신이 선택한 위치의 위험도를 알아보는 것으로 아래와 같은 입력 값을 가진다. 시야내의 적의 숫자, 자신주변의 아군의 수, 에너지 상태를 입력 값으로 한다. 이 3개의 입력 값으로부터 Safety값을 도출한다. 각각의 소속 함수들은 그림 9와 같다.

  Safety는 3개의 요소가 3개씩의 퍼지 값을 가지고 있고 추론규칙은 min-max method를 사용하였기 때문에 총 27개의 룰 셋을 가지고 있다.

라. 전략 Position

- Obstacle-Select, Attack-First, Safety과정에서 도출된 3개의 값으로 현재 위치의 전략적 위치 값을 얻을 수 있다. 전략 Position의 소속 함수는 그림 10과 같다.

위에서 정의한 Cascade-Fuzzy Logic 모듈은 다음과 같이 사용하였다. 예를 들면 Rule-Select과정에서 룰 값을 균등히 한다고 하였을 시 Obstacle-Select모듈에서는

(1) <엄폐물과 나와의 거리가 Close> 이고 <주위에 적의 숫자가 Normal> 이면 <Obstacle-Select은 Good> 이다.

(2) <엄폐물과 나와의 거리가 Medium> 이고 <주

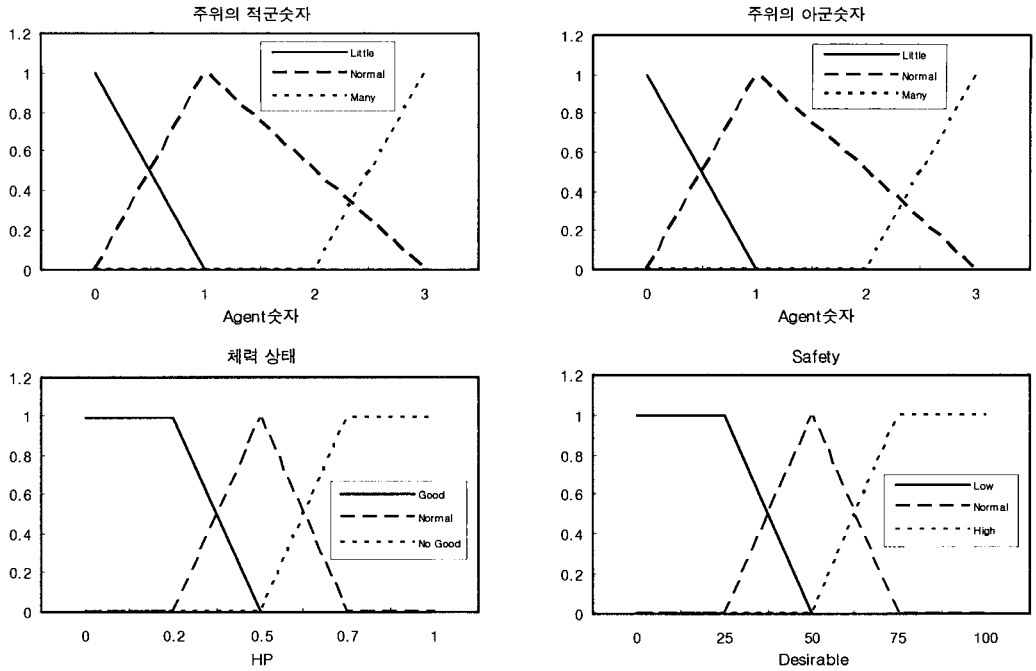

그림 9. Fuzzy Variable for Safety

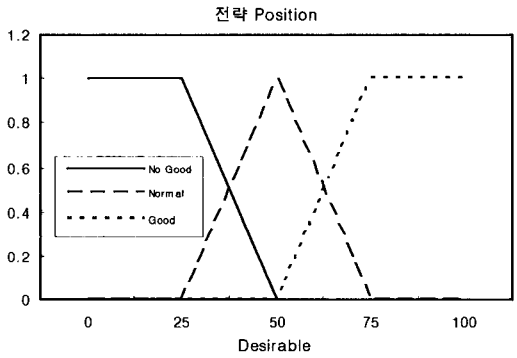

그림 10. Fuzzy Variable for 전략 Position

위에 적의 숫자가 Little> 이면 <Obstacle-Select은 Normal> 이다.

(3) <엄폐물과 나와의 거리가 Far> 이고 <주위에 적의 숫자가 Many> 이면 <Obstacle-Select은 No Good> 이다.

위와 같이 룰을 정의 하였고 무게중심 법으로 비퍼지 화하여 Obstacle-Select의 Desirable값을 도출하였다. Attack-First모듈에서는

(1) <Attack-Delay가 Now> 이고 <목표물과 사정거리의 관계가 Close> 이고 <지형의 종류가 안보임> 이면 <Attack-First는 같이 공격> 이다.

(2) <Attack-Delay가 Wait> 이고 <목표물과 사정거리의 관계가 Medium> 이고 <지형의 종류가 보임> 이면 <Attack-First는 먼저공격> 이다.

(3) <Attack-Delay가 Long> 이고 <목표물과 사정거리의 관계가 Far> 이고 <지형의 종류가 보임> 이면 <Attack-First는 공격 못함> 이다.

위와 같이 룰을 정의 하였고 무게중심 법으로 비퍼지 화하여 Attack-First의 Desirable값을 도출하였다. Safety모듈에서는

(1) <주위의 적군숫자가 Little> 이고 <주위의 아군숫자가 Many> 이고 <체력 상태가 Good> 이면 <Safety는 High> 이다.

(2) <주위의 적군숫자가 Normal> 이고 <주위의 아군숫자가 Normal> 이고 <체력 상태가 Normal> 이면 <Safety는 Normal> 이다.

(3) <주위의 적군숫자가 Many> 이고 <주위의 아군숫자가 Many> 이고 <체력 상태가 No

Good> 이면 <Safety는 Low> 이다.

위와 같이 룰을 정의 하였고 무게중심 법으로 비퍼지 화하여 Safety의 Desirable값을 도출하였다. 이렇게 도출한 3개의 모듈 값을 입력 값으로 하는 전략 Position모듈은 Obstacle-Select, Attack-First 와 Safety 모듈이 각각 3개의 퍼지 값을 가지므로 27개의 룰 값을 가지게 된다. 정의한 룰은

(1) <Obstacle-Select가 Good> 이고 <Attack-First가 공격 못함> 이고 <Safety가 High> 이면 <전략 Position은 Good> 이다.

(2) <Obstacle-Select가 Normal> 이고 <Attack-First가 같이 공격> 이고 <Safety가 Normal> 이면 <전략 Position은 No Good> 이다.

(3) <Obstacle-Select가 No Good> 이고 <Attack-First가 먼저공격> 이고 <Safety가 High> 이면 <전략 Position은 Normal> 이다.

위와 같이 룰을 정의 하였고 무게중심 법으로 비퍼지 화하여 전략 Position의 Desirable값을 도출한다. 이 값을 해당 Cell의 위치 값으로 사용하였다.

5. 실험 및 결과

제안한 AI플랫폼을 구현 및 AI개발에 적용해 보았다. 적용한 게임은 테스트를 위해서 Torque3D로 슈팅 게임을 제작하였다. Darwin은 모든 연결이 socket으로 되어있기 때문에 게임 환경에 독립적이라 Torque3D기반이 아닌 다른 환경에도 바로 적용을 할 수가 있다. 그림 11은 Darwin을 사용하여 적용한 그림이다. Darwin의 화면은 게임 화면과 Darwin을 제어할 수 있는 유저 인터페이스 창으로 이루어져 있다.

전략 모듈을 사용하여 개발한 에이전트가 일반적인 에이전트보다 보다 지능적으로 움직이는 것에 대한 결과는 다음과 같았다. 실험 환경으로서는 FSM을 기본구조로 하는 에이전트를 사용하였다. 이 에이전트는 A*알고리즘을 사용하여 길을 찾고 가장근거리에 있는 적을 목표로 설정하고 사정거리까지 쫓아가고 가변적으로 전후좌우로 움직여 조준을 방해하는 동작을 하는 일반적인 슈팅아케이드게임의 형식을 사용한다. 이런 에이전트를 사용 한 게임 환경으로 같은 수의 에이전트를 대결 시켰다.

그림 12, 그림 13, 그림 14, 그림 15 에서 알 수 있듯이 Darwin에서 제공하는 전략 모듈을 사용하면 엄폐물을 사용하며 보다 전략적인 위치를 선택하여 엄폐물 주위에서 싸우는 동선을 볼 수 있었다. 하지만 엄폐물이 존재하지 않는 맵일 경우 Attack-First와

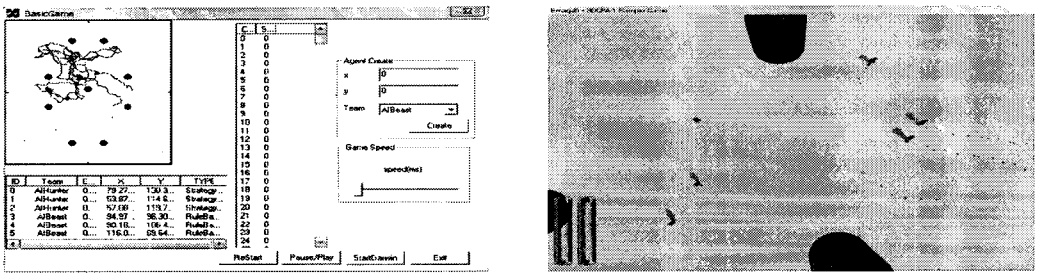

그림 11. Darwin의 화면

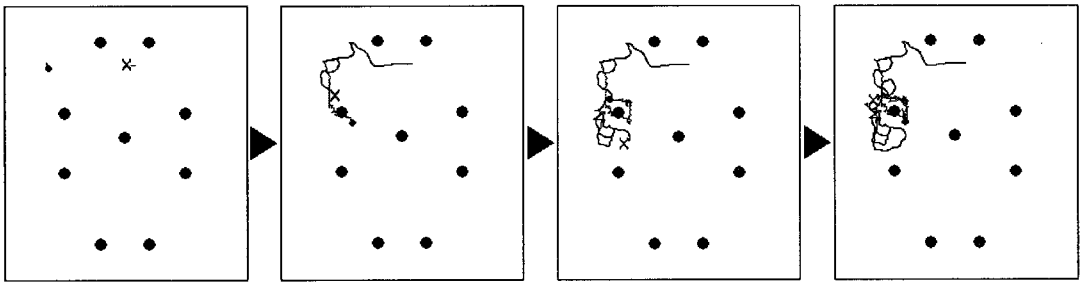

그림 12. Obstacle-Select의 룰 값을 우선시 했을 때 맵 환경과 위치 선택결과 (엄폐물 존재)

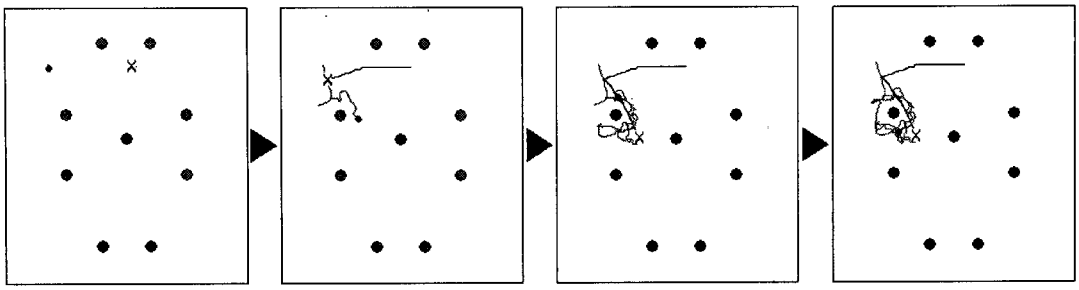

그림 13. 룰 값을 균등히 했을 때 맵 환경과 위치 선택결과 (엄폐물 존재)

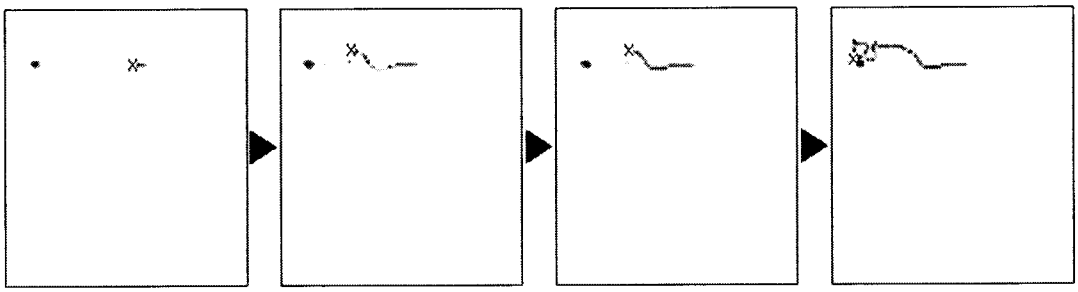

그림 14. Obstacle-Select의 룰 값을 우선시 했을 때 맵 환경과 위치 선택결과 (엄폐물 없음)

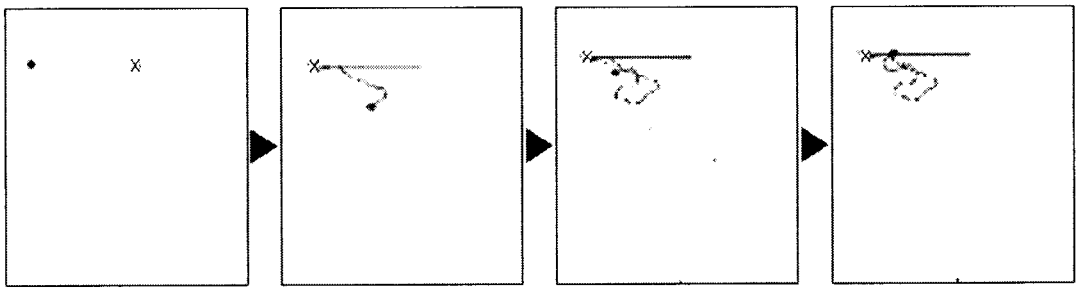

그림 15. 룰 값을 균등히 했을 때 맵 환경과 위치 선택결과 (엄폐물 없음)

Safety를 기준으로써 움직이기 때문에 엄폐물이 있는 맵과는 다른 동선결과를 보여주었다.

표 3과 표 4는 Obstacle-Select의 룰 값을 우선시 했을 때와 룰 값을 균등히 사용하였을 시에 제안된 전략 모듈을 사용한 에이전트와 사용하지 않은 에이전트와의 100전 결과 이다.

결과에서 알 수 있듯이 룰 값을 균등히 사용 하였을 시에는 전략 모듈이 엄폐물이 있을 때 좋은 결과를 보였다. 하지만 숨을 곳이 없는 엄폐물이 없는 맵에서는 상대와 마찬가지로 엄폐물에 숨으면서 공격할 수 없기 때문에 큰 차이를 보이지는 못하였다. 그리고 Obstacle-Select의 룰 값을 우선시 했을 시에는 전략 모듈이 엄폐물이 있을 때 룰 값을 균등히 사용한 것보단 좋진 않았지만 좋은 결과를 얻을 수 있었다. 하지만 엄폐물이 없는 맵에서는 엄폐물을 찾을

표 3. Obstacle-Select의 룰 값을 우선시 했을 때 100전 승패 결과

	승	패
엄폐물 존재	88	12
엄폐물 없음	31	69

표 4. 룰 값을 균등히 했을 때 100전 승패 결과

	승	패
엄폐물 존재	93	7
엄폐물 없음	64	36

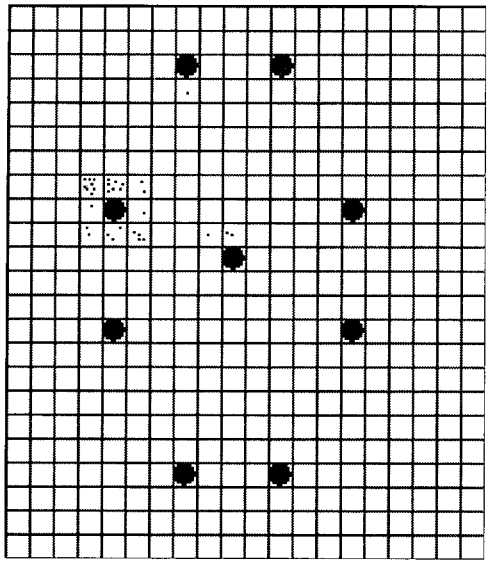

그림 16. Cell의 선택된 횟수

수 없어 정적인 모습을 보여주기 때문에 일방적으로 당하는 상황이 많이 발생하였다.

그림 16은 엄폐물이 있는 맵에서 전략 모듈을 사용하였을 때 한 번의 전투를 하는 동안 선택된 위치의 경향을 나타내고 있다.

6. 결 론

본 연구에서는 에이전트기반 게임 플랫폼의 독립성, 개발비용의 감소를 위한 다양한 인공지능 기법지원에 대해 요구사항을 분석하였다. 이것을 바탕으로 에이전트의 재사용을 가능하게 하고 게임로직과 에이전트의 독립적, 병렬적 개발을 가능하게 하는 게임환경에 독립적인 플랫폼인 Darwin을 제작 구현하였고 효율적인 사실적 에이전트개발 을 위해 전략적 위치를 찾아주는 퍼지 로직을 사용한 전략 모듈을 제안하였다. 이런 Darwin을 사용하여 학원에서 인공지능을 공부하는 학생들의 학습도구로써 또 게임회사에서 기획된 게임의 재미와 게임의 형태를 알아보는 테스트게임도구로써 또 게임개발 시 하나의 인공지능코드로 여러 게임플랫폼(PC-Online, 모바일환경)에서 구동이 가능한 도구로써 산업현장에서 쓰일 수 있을 것이라고 본다.

7. 추후연구

게임인공지능 개발에서 인공지능은 최대한 안정적으로 동작을 해야 한다. 그러므로 다른 게임인공지능 플랫폼(Robocode)과의 실시간 인공지능 업데이트의 안정성을 검사하여 업데이트 율이 유지가 되는지 알아보는 실험을 함으로써 Darwin플랫폼의 안정성에 대해 연구해야 할 것이다.

게임인공지능 개발을 용이하게 해주는 미들웨어의 수준을 테스트하여 개발에 도움이 되는지 연구해야 할 것이다. 실험 방법은 Rule base의 일반적인 지능을 가진 게임 에이전트와 본 연구에서 제안한 퍼지 로직을 사용하여 구현한 게임 에이전트끼리 대결을 하여 그 결과가 어떠한지에 대해 연구 하여야 할 것이다.

참 고 문 헌

[1] C. Crawford, *The Art of Computer Game Design*, Columbus: McGrawHill, 1984.

[2] M. Wooldrige and N. R. Jennings, "Intelligent agents: Theory and practice," *The Knowledge Engineering Review*, 10(2), pp. 115-152, 1995.

[3] http://www-903.ibm.com/developerworks/kr/robocode/robocode.html.

[4] R. Adobbati, "Second International Workshop on Infrastructure for agents," *MAS, and Scalable MAS*. 2001.

[5] "Gambots:A 3D Virtual World Test-Bed For Multi-agent Research," *In Proceedings of the Second International Workshop on Infrastructure for agents, MAS, and Scalable MAS*, 2001.

[6] Penny Baillie-de Byl, "Programming Believable Characters for Computer Games," *Charles river media*, 2004.

[7] Chen, G. *Introduction to fuzzy sets, Fuzzy Logic, and fuzzy control systems*, CRC Press, 2001.

[8] W. Pedrycz and F. Gomide, *An Introduction to Fuzzy Sets*, Cambridge: MIT Press, 1998.

이 원 희

2005년 안양대학교 디지털미디어공학과 학사 졸업
2007년 중앙대학교첨단영상대학원 영상공학과 석사 졸업
관심분야 : 인공지능, 게임

김 원 섭

2006년 2월 한국기술교육대학교 인터넷미디어공학부 졸업(공학사)
2006년 3월~현재 중앙대학교 첨단영상대학원 영상공학과 석사과정
관심분야 : 웹, 컴퓨터 비전, 컴퓨터게임

김 태 용

1986년 2월 한양대학교 전기공학과(공학사)
1988년 2월 한양대학교 전자통신공학과(공학석사)
1998년 2월 포항공과대학교 컴퓨터공학과(공학박사)
1988년 3월~1999년 2월 한국통신 운용연구단 연구원
2003년 4월~현재 한국컴퓨터게임학회 총무이사
2000년 1월~현재 중앙대학교 첨단영상대학원 영상공학과 부교수
관심분야 : 영상통신, 영상처리, 컴퓨터비전, 컴퓨터게임