論文2001-38CI-3-3

동적 환경 내에서의 자율 에이전트에 의한 다양한 행위의 설계 및 구현

(Design and Implementation of Diverse Behaviors of Autonomous Agents in Dynamic Environment)

朴 亨 根 *, 朴 宗 熹 *

(Hyung-Keun Park and Jong-Hee Park)

요 약

에이전트의 정의와 구현에 관한 연구는 가상현실기술을 이용한 몰입형 교육 시스템 등의 구성에 있어서 매우 중요한 부분을 차지한다. 본 논문에서는 동적인 환경에서 에이전트의 행위의 효율적인 정의와 구현 방법을 제시하되 기존의 연구와 달리 *행동의 다양성*의 표현에 초점을 맞추고자 한다. 먼저 장이론에 기반하여 공간객체를 정의한다. 장이론을 도입하여 공간객체를 정의함으로써 공간 객체에 대한 분류적 접근이 아닌 구조적인 접근이 가능해 지고, 공간객체의 동적인 생성 및 소멸과 전체적인 시공간 상황의 변화를 유도할 수 있게 된다. 다음으로 동적인 planning이 가능하도록 에이전트의 행위를 복합행위와 단위행동으로 구분한다. 마지막으로 에이전트와 관련된 객체의 관계에 따라 지식 베이스를 에이전트와 관련 객체들에 분산한다. 이를 통해 에이전트의 지식베이스의 효율적인 관리와 동적인 환경의 구축이 가능하다. 공간객체 내에서의 에이전트의 이동에 관한 상황을 구현함으로써 장이론에 기반한 공간객체의 정의와 행위의 복합행위와 단위행동으로의 구분, 그리고 에이전트와 관련 객체간의 지식 베이스의 분산의 효율성을 검증하였다.

Abstract

The design and implementation of agents is an essential part of the development of immersive types of tutoring systems using virtual reality. This paper proposes several effective mechanisms for the design and implementation of agents. Unlike existing researches we focus on accommodating *diversity of agents' behavior* in the proposed mechanisms. First, we define the space object based on the field theory. The introduction of the field theory allows us to approach the space objects in a structural manner rather than by their classification. We can also achieve dynamic genesis and extinction of the space objects, and derivation of overall changes in spatio-temporal situations. Second, we classify the behavior of agents into composite behaviors and primitive actions in order to achieve its dynamic planning. Finally, we distribute the knowledge among agents and their associated objects according to their interrelations. By this distribution, we can handle the otherwise prohibitively large amount of knowledge related to agents' behaviors and construct a dynamic environment. By implementing a situation with agent's navigation across a composite space object, we demonstrate the effectiveness of these schemes presented above.

* 正會員, 慶北大學校 電子工學科
(School of Electronic & Computer Engineering, Kyungpook National University)

※ 본 연구과제는 정보통신부의 대학정보통신연구센터 육성 지원사업의 예산으로 수행된 것임을 밝힙니다.
接受日字:2000年2月21日, 수정완료일:2001年4月20日

Ⅰ. 서 론

가상 현실은 컴퓨터가 만들어낸 동적인 환경 속에서 인간이 자신의 감각을 좌우하고 컴퓨터가 생성한 환경과 직관적으로 상호 작용할 수 있도록 하는 기술이며 이는 교육 시스템에서 중요한 응용 기술로 사용되어져 학습자가 가상 현실 상황에서 간접적으로 실세계를 경험하며 그에 대한 반응을 행함으로서 기술 향상을 얻을 수 있게 된다[1~4]. 가상현실을 이용한 몰입형(immersive) 교육 시스템[4]을 구성함에 있어서 중요하게 여겨지는 것은 가상현실 상황에 존재하는 에이전트의 정의와 구현이다. 에이전트에 대한 정의는 보는 시각에 따라 다양하게 제시되어질 수 있으나 이를 공통적인 입장에서 보면 사용자를 대신하여 복잡하고 역동적인(dynamic) 환경에서 일련의 주어진 목표(goal)를 수행하는 하나의 시스템이라고 볼 수 있다[5~7]. 에이전트의 특성은 크게 세 가지로 구분되어질 수 있는데, 인식되어진 환경에 따라 목표가 성공적으로 수행되어질 수 있도록 행동을 함에 있어서 자기 스스로 판단하는 자율성(autonomy), 경험을 통해 학습되어진 지식을 바탕으로 좀 더 나은 행동양식을 가지게 되는 적응성(adaptation), 그리고 협동성(cooperation)이다[7]. 즉 에이전트는 인접한 환경과 상호 작용하여 그 속에서 학습을 통하여 스스로를 발전시켜나가며 주어진 과제에 대해 자신의 능력으로 대처하여 최선의 방법을 찾아야 한다. 에이전트가 상호 작용하는 환경은 자신이 위치한 공간과 그 공간 속에 존재하는 다른 에이전트들, 그리고 다른 객체들로 구성되어질 수 있다. 그리고 일반적으로 에이전트는 환경을 인식하기 위한 인지요소(sensors), 인식되어진 요소로부터 다음의 행위를 결정하는 결정요소(decision-making), 그리고 결정되어진 행위로서 환경에 대해 반응하는 구동요소(effectors)를 최소 구성요소로 하며, 이들 요소 사이에서 정보처리 능력이 매우 중요시 여겨진다[9]. 이러한 구성요소들 중에서 환경에 대한 반응을 결정하는 요소의 경우 단순한 인식의 결과에 대한 반응이 아니라 에이전트에게 부여되어진 과제(task)에 따라 이를 성공적으로 수행하기 위해 미리 주어진 지식과 자신이 학습한 지식 그리고 인식되어진 환경에 따라 최선의 방법을 스스로의 판단하는 자율성이 강조되는 자율 에이전트(autonomous agent)를 구현하고자 한다.

본 논문에서는 몰입형 교육 시스템에서 필요한 다양한 상황과 그에 따른 자율 에이전트의 자연스런 행동을 구현하기 위해 몇 가지 설계 방법론을 제시한다. 몰입형 교육시스템의 학습의 무대로서의 다양한 상황들은 자연현상이나 인간의 행동양태에 의해 대부분 발생한다. 본 논문은 그 중에서 특히 인간의 의도나 판단에 따라 발생 및 전개되는 상황들의 구현에 초점을 맞춘다. 이를 위해서는 수많은 상황들 속에서 중심적 역할을 하는 인간 즉 에이전트의 행동이 자연스럽게 표현되어야 한다. 에이전트가 자연스런 행동을 할 수 있게 되려면 상응하는 인간의 판단 및 행동양식을 설계에 반영할 수 있어야 한다.

첫째, 에이전트는 주변 환경으로부터 영향을 받고 그에 대해 반응하게 된다. 이러한 환경과 에이전트간의 상호작용의 중요성은 지금까지의 여러 연구에서 중시되어져 왔다[8~12]. 그러나 실제 이런 환경의 체계적 구조화나 환경과 에이전트사이의 상호작용에 관한 명확한 방법론이 제시되지 않고 있다. 본 논문에서는 에이전트와 환경의 상호작용을 체계화하기 위해 장이론(field theory)[8]을 응용하여 공간객체를 정의하고 환경을 체계화한다[13]. 공간환경은 물리적 객체를 기반으로 하여 구성되나 논리적 개념에도 유추적용될 수 있도록 한다. 이와 함께 특정 공간만이 가지는 규칙을 정의할 수 있게 함으로써 다양한 상황의 구현이 가능해질 수 있다.

둘째로, 에이전트가 가지는 행동은 각각의 작은 행동단위를 나타내는 기본행동(primitive action)의 집합체로 나타내어질 수 있다[3]. 에이전트의 무수한 형태의 행동들을 구성하는 기본적인 에이전트의 행동을 정의하고 이들의 여러 가지 조합에 의해 다른 행동들이 유발됨을 보이고자 한다.

셋째로, 각 에이전트의 구체적 행동은 주변 환경요소들의 특성에 따라 적합하게 조절된다. 이에관해 기존의 에이전트의 설계에 있어서는 에이전트가 모든 주변 상황을 인식도록 하고 그에 대한 지식도 모두 자신의 기억구조에 저장하여야 하였다. 본 논문에서는 환경에 따른 에이전트의 행동양식을 추상적 지식만 에이전트 자신의 지식베이스에 저장하고, 구체적 행동에 관한 지식은 해당 객체들의 지식베이스에 분산 저장한다. 실제적 행동은 이러한 분산된 두 가지 지식을 메시지 전달 방법을 통하여 조합함으로써 구현된다. 이렇게 함으로써 주변 상황에 따른 에이전트들의 다양한 행동을 구현하

는데 필요한 정보를 최소화할 수 있게 한다. 이러한 방법은 에이전트들이 구체적 행동을 함에 있어서 객체들의 실제적 특성을 무의식적으로 파악하고 적응하기 때문에 에이전트간의 차이를 무시할 수 있기 때문이라는 관찰에 근거한다.

이러한 방법론들은 언어교육등[14]을 위한 몰입형 교육을 위한 사이버환경을 체계적으로 구축하기 위한 방안들이다. 이는 개별 에이전트가 공간상에서 경로의 찾기등의 개별적 목적을 효율적으로 달성하기 위한 시스템들[15,16]과 근본적으로 다른 점이다. 구체적으로 실세계에서의 다양한 상황들을 자연스럽게 표현하고 그러한 상황에 처한 인간들의 행동 양태를 실제와 유사하게 나타낼 수 있는 사이버환경을 제공하기 위한 방안들이다. 이러한 자연스런 사이버 환경의 구현을 위한 수많은 요소중에서 구조의 논리적 확장성과 필요한 정보의 효율적 처리가 본 연구의 목표이다. 논리적이란 의미는 상황의 시각적인 측면에 관한 표시에만 국한하지 않고 상황의 전개를 지배하는 법칙이나 원리에 구현의 초점을 둠을 의미한다. 비록 공간적 이동이라는 상황만을 중심으로 논의를 전개하더라도 상황구축에 필요한 지식구조나 상황에 따른 판단을 위한 절차는 논리적 확장성을 가진다. 즉 만유인력등의 자연적인 인과관계뿐만 아니라 인간간의 감정에 의해 촉발되는 관계등의 추상적 관계들에도 적용가능한 형태로의 확장성을 지니게 된다. 이에 관해 첫째 설계 방법론은 기존의 연구들에서 경우에 따라 일정한 원칙없이 설계되던 다양한 환경과 에이전트와 환경간의 상호작용을 체계적으로 구조화하는 방안이다. 상황의 다양성은 소요되는 정보의 분량에 의해 제한되는 경우가 많다. 따라서 상황의 다양성을 가능하게 구현하는데 필요한 방대한 정보량을 최소화하는 방안은 본 사이버 환경의 구현의 성패를 결정짓는 실질적인 요소의 하나다. 이에 관해 둘째와 셋째의 방법론들은 에이전트와 환경사이의 접촉에서 벌어지는 무수한 행동양식들을 효율적으로 구현할 수 있게 해주는 방안들이다.

Ⅱ. 관련 연구

컴퓨터를 이용한 애니메이션(computer animation) 분야에서는 오랫동안 Path model이 지배적으로 사용되어져 왔다[15]. 이는 객체의 행위가 일련의 순서에 따라 발생하는 것처럼 이를 이미 준비되어진 프레임의 열거에 의해 컴퓨터 상에 표현하는 방법이다. 이러한 Path model은 고정적 환경에서 자신만의 행동을 규정하는 방법으로는 유효하다. 그러나, 동적인 환경에서 객체가 환경에 의해 지속적으로 자극과 제약을 받게되는 경우에는 그러한 독립적 행위만으로는 다양한 상황과 예기치 못한 상황을 표현할 수 없게 된다. 동적인 환경은 각기 다른 방향으로 행동하는 객체들과 불특정한 시간에 발생되어지는 사건(event)들을 포함하고 있기 때문이다. 이러한 기술의 제약점을 극복하고자하는 노력으로 Sensor-Effector model, Rule-based model, Predefined model 등이 새로이 연구되어지고 있다[15].

Sensor-Effector model은 감지요소, 구동요소, 그리고 그들 사이를 연결하는 신경망(Neural network)으로 구성되어진다. 이 모델은 성공여부는 실제적인 신경망이 어떻게 작동되는지 이해하느냐의 여부에 달려있다고 할 수 있다. 하지만 지금까지 오랫동안의 연구에도 불구하고 실제 생물의 신경망의 동작 원리는 연구과제로 남아있다. Rule-based model은 감지요소와 구동요소를 연결하는 방법으로 입력을 출력으로 연결(mapping)하기 위한 행위규칙을 이용한다. Rule-based model의 중앙 제어 장치(Central Control Process)에서 사용되어지는 행위 규칙들의 집합은 객체가 접한 환경으로부터 가장 적절한 행동을 찾는다. 그러나 동적환경에서는 구축비용이 엄청나게 소요되며, 행동의 각 단계마다 가능성을 가지는 모든 행위를 찾기 위해 수많은 규칙의 결정트리(Decision tree)를 탐색해야 한다. 또한 조그만 환경의 변화라 하더라도 이를 결정트리에 반영하는 것은 매우 어렵다. 또 다른 방법인 Predefined Environment model은 동적인 환경이 아닌 미리 정해진 환경만을 대상으로 하는 모델이다. 이 모델에서는 객체의 애니메이션 이전에 환경이 미리 결정되어지므로 객체의 행동은 그 이전의 지식을 바탕으로 결정되어지는 한계가 있다.

능동객체 시스템의 하나인 Field-to-Field model은 물리적 영향력에 관한 장이론을 가상현실에 확장시켜 이를 시각적으로 표현하고, 2차원의 가상환경에서 이동하는 객체들이 장애물을 피하는 행위의 구현에 적용하고 있다[13]. 그러나 Field-to-Field model은 에이전트의 외부환경 중에서 다른 에이전트와의 영향력만을 고려하여 충돌방지에 적용함으로써 가상 시공간 상황에서 에이전트의 다양한 공간이동과 그에 따른 행위의 변화를 표현하는데 있어서는 근본적 한계를 가지고 있다.

또한 물리적 영향력에 치중하여 에이전트가 설정한 공간이 가지는 규칙 등의 논리적 영향은 고려되지 않고 있다. 본 논문에서는 이러한 한계를 극복하기 위해 에이전트가 가지는 환경의 한 요소인 공간객체에 적용이 가능하도록 장이론을 확장시킨다.

에이전트의 공간이동에 관련된 행위의 Planning에 대한 기존의 연구 중에서 Object Geometric Hierarchy model(OGH)을 이용한 방법이 본 논문에서 제안된 공간객체의 정의를 통한 Space hierarchy와 유사성을 보여준다[16]. 그러나 OGH model은 공간객체와 사물객체와의 구별을 두지 않아 가상공간상에 존재하는 에이전트의 공간이동에 관한 행동에 대해서 모순점을 가진다. 그리고 상위공간객체가 여러 개의 내부공간객체로 분할되어진 환경의 경우, 내부공간간의 공간이동의 논리적으로 설명이 불가능하다. 또한 상위공간으로의 이동에 대해서도 OGH model은 부분적인 고려에 기초한 Planning에 그치게 된다.

Ⅲ. 행위 표현을 위한 지식 구조의 설계

1. 장이론(Field theory)을 이용한 공간객체의 설계

환경과 에이전트간의 관계: 하나의 에이전트가 나타내는 행위의 근본은 에이전트가 외부로부터의 자극상태와 해당 시기에 자신이 가지는 상태에 달려있다. 이는 외부로부터의 동일한 자극에 대해서 현재 에이전트가 속해 있는 물리적 환경(공간)이나 자극의 성질의 차이 그리고 에이전트의 내부의 상태 등에 따라 파악되는 인식에 차이가 생기게 되고 그에 따른 다양한 행위 그리고 사건들이 발생하게 된다[8]. 즉, 에이전트의 행위(B)란 에이전트의 상태(P)와 그를 둘러싼 환경(E)의 함수(F)로 될 수 있다.

$$B = F(P, E)$$

위에 제시된 공식에서 에이전트의 상태(P)와 에이전트가 속한 환경(E)의 관계도 또한 독립적이 아니다. 환경은 에이전트의 행위뿐만 아니라 상태에도 영향을 미쳐 환경에 대한 에이전트의 인식에 변화를 가져오게 되며, 에이전트의 상태 변화에 따른 행위 또한 환경에 영향을 미치게 되므로 에이전트의 상태와 그가 처한 환경은 서로 상호 영향을 주고받게 된다. 즉,

$$P = F(E),\ E = F(P)$$

인 관계가 성립된다. 결론적으로 에이전트의 행위는 그가 위치한 환경과 자신 내부의 상태에 따라 변화하게 되고, 이러한 요인으로 인해 발생된 행위는 또 다시 환경의 변화를 가져오게 되는 요인으로 작용한다.

공간객체의 설정: 가상 현실 상에 존재하는 에이전트에게 있어 주위환경은 그림 1에서 보이는 것과 같이 관련된 에이전트들, 에이전트에 관계된 사물객체들, 그리고 에이전트가 위치하게 되는 공간들로 구성되어질 수 있다. 이렇게 구성되어진 에이전트 객체, 사물객체, 공간 등과의 상호작용을 통해서 에이전트는 다양한 상황을 인식하며 이에 대해 반응한다[9,11].

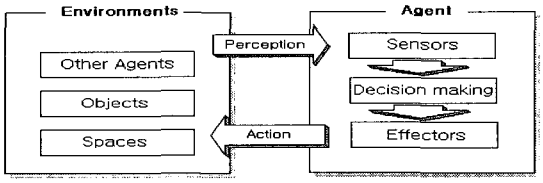


그림 1. 환경과 상호 작용하는 에이전트
Fig. 1. Agent interacting with environments.

본 논문에서는 이러한 환경의 구성 요소들 중에서 먼저 공간이라는 객체에 대해 장이론을 적용한다. Lewin의 정의를 따르면 장은 개인의 생활공간(life space)이다[8]. 그리고 장이론은 인과관계를 분석하고 과학적 구성체를 종합하는 학문으로 에이전트와 환경사이의 문제를 다룸에 있어서 장소, 인지구조, 힘 등의 개념을 다루는데 중심 되는 기본 개념으로 제시되었다[8]. 장이론을 적용하여 에이전트의 환경의 하나의 요소인 공간객체를 정의함으로써 가지게 되는 구체적 효과는 다음과 같다. 첫째로 공간객체에 대해 자의적인 분류에 의존하는 방법이 아닌 구조적인 방법으로 접근이 가능하게 된다. 즉, 공간 객체가 공간계층구조(Space Hierarchy)를 이루는 상대적 상위공간객체와 하위공간객체로 구분되어지며, 하위공간객체는 상위공간객체와 영향을 주고받는 관계로 정의된다. 둘째로 동적인 환경의 구현에 효율성을 제고할 수 있게 된다. 공간객체는 주어진 시공간 상황에서 동적으로 생성과 소멸이 가능하다. 하나의 공간객체의 생성 및 소멸은 상위공간객체와 하위공간객체에 장의 형태로 영향을 미치게 되고 전체적인 시공간 상황의 변화를 유도할 수 있다. 그리고 이는 에이전트에 의한 전체적 상황의 분석의 출발점으로 제시되어질 수 있다. 이러한 장에 기초한 방법은 다른 에이전트와의 상호작용에 관련된 심리적 장에

도 적용[11]될 수 있으나, 본 논문에서는 공간객체에만 중점을 두어 객체의 구조를 정의하고 이의 특성에 따른 에이전트의 행위 방식의 변화에 대해서만 논하기로 한다.

공간객체의 구조: 하나의 객체가 상황에 등장하게 될 경우 이 객체에 의해 발생되어지거나 유도되어지는 장은 크게 두 가지로 볼 수 있다. 즉 영향장(Stimulative field)은 객체가 가진 영향력이 미치는 공간이며, 인식장(Sensing field)은 객체가 다른 객체의 영향을 받아들일 수 있는 공간이다[13].

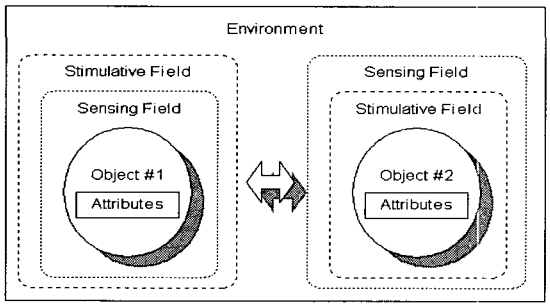


그림 2. 객체가 가지는 영향장과 인식장
Fig. 2. Stimulative and Sensing fields of objects.

그림 2는 객체가 상황에 등장할 경우 그의 속성(attributes)에 의해 유도되는 영향장과 인식장을 도식적으로 나타낸 것이다. 영향장과 인식장의 크기는 attributes를 매개변수로 하여 주어지는 수학적 함수로 나타내어질 수 있으며 상황에 따라 변화가 가능하다. 이와 같은 장의 분류를 공간객체에 적용할 경우, 영향장은 그 공간의 내부로 설정되어질 수 있으며 인식장은 공간 내부를 포함하여 그 공간이 위치한 상위 공간객체가 될 수 있다. 예를 들어 건물 내부의 하나의 공간객체인 사무실 환경은 사무실 내부에 있는 모든 객체에 영향을 미칠 수 있고, 한편 상위 공간객체인 건물 내부의 규칙, 더 나아가 건물이 위치한 지역의 영향, 그리고 내부의 객체에 의해 영향을 받게 된다. 이러한 방법에 의해서 가상공간 상에 설정되어지는 하나의 공간객체는 생성 시에 이미 존재하는 상위 공간객체의 특성과 규칙을 인식장에 의해 상속받게 되고, 구현 시에 공간객체 내에 등장하게 되는 에이전트나 객체에게 영향장을 통해 작용하게 된다.

공간객체의 특성: 공간객체는 생성 시 어떠한 객체로부터 유도되어 생성되었는지에 따라 크게 단일 유도 공간객체(single_object space)와 복합 유도 공간 객체(multiple_object space)의 두 가지로 구분되어질 수 있다. 단일 유도 공간객체는 하나의 사물 객체 또는 에이전트가 상황에 등장할 경우 이에 연관되어 나타나는 공간 객체로 객체 자신이 차지하는 물리적 공간 또는 상대적으로 객체의 상·하·좌·우에 위치하게되는 공간이 될 수 있다. 복합 유도 공간객체는 여러 개의 사물 객체가 집합하여 하나의 공간객체를 이루는 것이다. 다음의 그림 3 에서 제시된 환경에서 살펴보면 공간객체 House_1은 8개의 CBrick객체(Brick_1~Brick_8)와 3개의 CDoor객체(Door_1~Door_3)로 이루어진 복합 유도 공간객체이고, Desk_2에 의해 유도되어지는 공간객체 On_Desk_2, Under_Desk_2, Inside_Desk_2 등은 단일 유도 공간객체이다. 단일 유도 공간객체는 원 객체가 이동할 경우 함께 이동하는 특성을 가진다. 모든 공간객체는 상위 공간객체와 내부 공간객체와의 상대적 위치 관계를 가지므로 이러한 상관관계로부터 공간계층구조를 정의할 수 있다. 이는 에이전트가 위치 이동에 관계한 행동의 Planning에 유용한 자료로 활용되어진다.

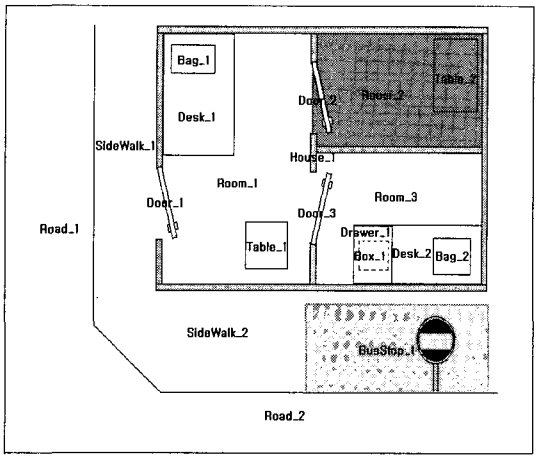


그림 3. 에이전트 작업을 위한 환경
Fig. 3. Working environment 1 for agents.

우선 그림 3에 설정된 환경에 대한 공간계층구조를 정의하고, 이 환경에서 위치 이동을 하는 에이전트의 Planning 과정을 논하기로 한다. 먼저 그림 3에 포함된 공간 계층 구조를 정의하면 그림 4와 같다. 화살표의 방향은 상위 공간객체를 향하게 되고 반대 방향은 내부 공간 객체임을 나타낸다. 실선으로 그려진 원은 복

합 유도 공간객체를 나타내고, 점선으로 그려진 원은 단일 유도 공간객체를 나타낸다. 에이전트가 공간을 이동함에 있어서 요구되어지는 Planning의 과정에 있어서 자신이 속한 환경의 공간계층구조를 이용한다. 에이전트는 공간객체와의 메시지교환을 통해 이러한 포함관계를 인식하게 된다.

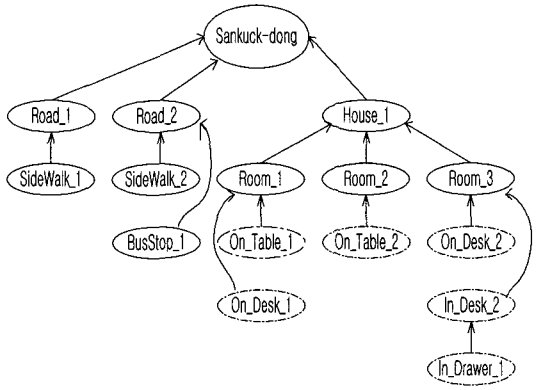


그림 4. 공간 계층구조
Fig. 4. Space hierarchy.

Planning은 어떤 행동에 대해 그 행동을 실행할 기본적인 행동들을 계획함으로써 실행된다. 또한 Planning은 단순히 실행하는 것뿐만 아니라 객체의 행동을 역사적 사실로 기억하기 위해 행동을 저장할 수 있는 구조로 표현되어야 한다. 본 논문에서는 에이전트의 공간이동을 위해 에이전트의 인식구조 속에 자신이 이동해야할 공간객체에 대한 리스트를 작성하고 이를 순차적으로 수행한다. 다음은 이러한 리스트를 Planning하는 과정이다.

1) Step 1 : 에이전트 자신이 속한 공간객체를 출발점으로 하여 공간계층구조를 탐색하여 자신이 이동하고자 하는 공간객체까지의 path를 공간이동 리스트에 작성한다. 그림 3의 환경에서 에이전트가 공간객체 Room_1에서 공간객체 BusStop_1까지 이동하는 경우에 생성되어지는 초기 list는 다음과 같다.

list = (Room_1, House_1, SanKuck-dong, Road_2, BusStop_1)

2) Step 2 : 생성되어진 리스트를 탐색하여 참조중인 특정한 공간객체에서 다른 공간객체들로의 출구가 존재할 경우 중간과정에 포함되어진 이러한 공간객체들로의 이동을 삭제한다. 이는 에이전트가 인접공간객체(Adjacent space object)를 인식함으로서 공간이동 리스트가 불필요하게 증가되는 것을 막고, 공간 이동에 대한 효율성을 부여하기 위한 방법이다. 그림 3의 환경에서 에이전트가 Room_1에서 SideWalk_1으로 이동할 경우 규칙1에서 얻어지는 리스트는 (Room_1, House_1, Sankuck-Dong, Road_1, SideWalk_1) 이다. 그러나 이러한 결과 리스트와 Room_1이 가지는 공간특성에서 살펴보면 Room_1은 Room_2, Room_3, SideWalk_1, Road_1으로의 출구를 가진다. 따라서 실제적으로 재조정된 리스트 (Room_1, SideWalk_1) 을 얻게 된다.

3) Step 3 : 공간이동 리스트 내에서 상위공간객체로의 이동이 2단계이상 계속되고, 1단계 상위공간객체가 여러 개의 내부공간객체로 분할되어 나누어져 있으며, 1단계 상위공간객체의 내부공간객체 중 하나가 2단계 상위공간객체로의 출구정보를 가지고 있을 경우, 해당 공간객체를 에이전트가 위치한 공간객체의 다음의 이동공간으로 이동 리스트에 포함시킨다. 이는 1단계 상위공간객체가 여러 개의 내부공간객체로 완전하게 분할된 경우, 1단계 상위공간객체의 직접적인 이동이 불가능하고 내부공간객체 중 일부가 2단계 상위공간객체로의 직접적인 출구정보를 가지기 때문이다. 이러한 문제점을 해결하기 위해 에이전트는 1단계 상위공간객체의 다른 내부공간객체로의 이동이 선행되어져야 한다. 그림 3의 환경에서 공간이동 리스트가 (Room_3, House_1, Sankuck-dong) 으로 주어졌을 경우, 공간객체 House_1은 공간객체 Room_3의 상위공간객체인 동시에 공간객체 Sankuck-dong의 하위공간객체이다. 이 경우 에이전트가 House_1을 벗어나 Sankuck-dong으로 이동하기 위해서는 해당 출구정보를 바탕으로 하면 에이전트가 House_1의 하위공간객체인 Room_1로의 이동이 선행되어야 한다. 선행적인 작업을 에이전트에 부과하기 위해 공간이동 리스트를 수정할 필요가 생긴다. 수정되어진 리스트는 (Room_3, House_1, Room_1, House_1, Sankuck-dong) 과 같다.

4) Step 4 : 다음 단계의 이동 대상 공간객체가 에이전트 자신이 위치한 공간객체의 상위 공간객체인 동시에 여러 개의 내부공간객체로 분할되어 있으며, 계속해서 그 내부공간객체의 하나로 에이전트가 이동하게 될 경우, 즉 하나의 상위공간객체에 대해 서로 다른 내부공간객체가 리스트상에 있어서 앞뒤로 위치할 경우, 상위공간객체로의 이동을 생략하고 리스트를 재정렬한다. 이는 규칙3과 마찬가지의 경우로 상위공간객체가 여러 개의 내부공간객체로 완전히 분할되어 있을 경우 상위

공간객체로의 이동이 불가능하게 된다. 계속해서 내부 공간객체중의 하나로의 이동이 이루어진다면 문제점은 상위공간객체에 포함되어진 모든 내부공간객체들의 인접 공간객체로의 출구 정보를 이용한 탐색을 통해서 해결되어진다.

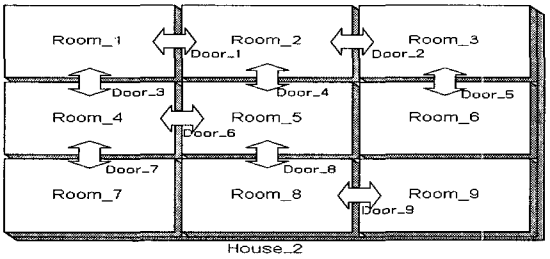


그림 5. 에이전트 작업을 위한 환경 2
Fig. 5. Working environment 2 for agents.

그림 5에서 보면 공간객체 Room_1에서 공간객체 Room_9으로 이동할 경우 에이전트는 (Room_1, House_2, Room_9) 의 초기 공간이동 리스트를 가진다. 그러나 공간객체 House_2는 Room_1의 상위공간객체인 동시에 여러 개의 내부공간객체로 나뉘어져 있으므로 Room_1에서 House_2로의 이동은 불가능하다. 즉 리스트에서 상위공간객체 House_2를 생략하고 이를 다른 내부공간객체들의 나열로 대처할 필요성을 가진다.

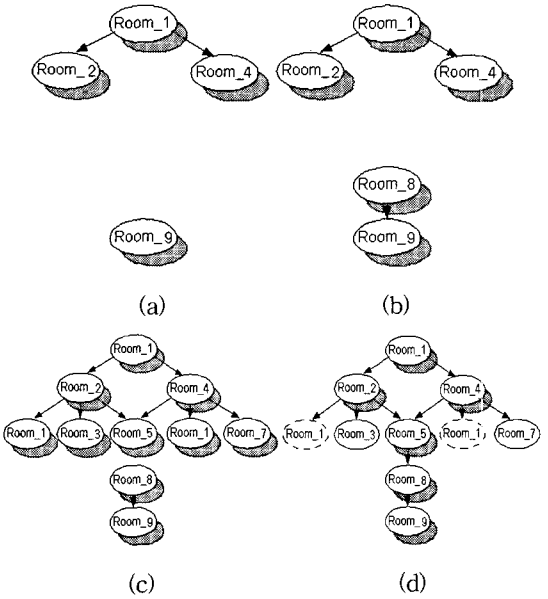


그림 6. 동일 상위공간객체를 가진 하위공간객체내의 공간이동
Fig. 6. Navigation across CSpace-objects with a common superior CSpace-object.

공간객체와 도착 공간객체를 각각 출발노드(Starting Node)로 설정한 뒤, 각각의 출발 공간객체에서 다른 공간객체로의 출구를 넓이우선탐색(Breadth-first Search)으로 탐색하여 트리를 확장시켜 나간다. 확장의 과정에서 이미 참조되어진 노드(공간객체)는 생략한다. 출발 공간객체에서 확장되어진 트리와 도착 공간객체에서 확장되어진 트리가 특정 공간객체에서 만날 때까지 계속해서 확장을 한다. 출발 공간객체로부터 도착 공간객체로의 통로(path)가 성립되면 이를 공간이동 리스트에 포함시킨다. 초기 list (Room_1, House_2, Room_9) 에서 탐색과정을 거쳐 재정렬된 최종 공간이동 리스트는 (Room_1, Room_2, Room_5, Room_8, Room_9) 로 변환되어진다.

위의 단계들을 종합적으로 이용하여 그림 3에 주어진 환경에서 에이전트가 공간객체 Room_3에서 BusStop_1로 이동하는 과정에 대한 전체 planning과정을 표현하면 그림 7과 같다.

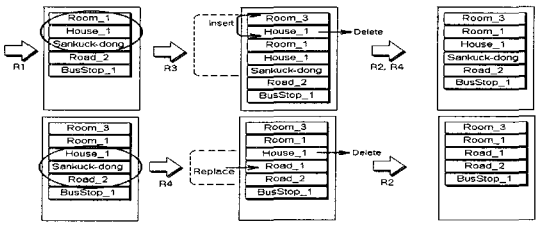


그림 7. 공간이동에 관한 Planning과정의 예
Fig. 7. Planning example for space navigation.

본 공간객체 설계와 공간 이동에 관한 방법론을 고찰해 보면, 에이전트의 공간이동에 관련된 행위의 planning에 대한 기존의 연구 중에서 Object Geometric Hierarchy Model (OGHM)을 이용한 방법이 본 논문에서 제안된 공간객체의 정의를 통한 Space hierarchy와 유사성을 보여준다[16]. 그러나 OGHM은 본 논문과 비교하여 몇 가지의 문제점을 가진다. 첫째 OGHM은 하나의 노드에 객체를 위치시키고 해당 객체를 포함하는 객체를 상위레벨에 위치시키고 또 해당객체가 포함하는 다른 객체들을 하위레벨에 두어 이를 확장시킨 모델로서 개념적으로는 본 논문에서 제안된 공간객체 계층구조와 유사하다. 그러나 OGHM은 공간객체와 사물객체와의 구별을 두지 않기 때문에 가상공간상에 존재하는 에이전트의 공간이동에 관한 행동에 대해서 불합

리점을 가진다. 일례로 서재에 있는 책상 객체 위에 위치한 컵 객체를 집어드는 행위를 표현할 경우 OGHM은 Move(Study, Desk) > Move(Desk, Cup_2) 로 표현한다. 그러나 실제는 에이전트가 책상 객체로 이동하고 다시 컵 객체로 이동하는 것이 아니라, 책상 객체 앞으로 옮겨가서 손을 뻗어 컵 객체를 잡는 행동으로 표현되어져야 한다. 따라서 본 논문에서와 같이 공간객체와 사물객체의 구분이 필요하게 된다. 둘째 본 논문에서 제안된 것처럼 상위 공간객체가 여러 개의 내부 공간객체로 분할되어진 환경의 경우, 내부공간 사이의 공간이동은 논리적인 설명이 불가능하다. 또한, 상위공간으로의 이동에 관해 본 논문에서 제안된 모델은 planning 과정에서 상충되지 않는 다른 공간으로의 이동으로 대체되나 OGHM은 본 논문의 규칙 1과 같은 수준의 planning 에 그치게 된다.

2. 객체와 에이전트간의 상호작용 방법의 설계

단위 행동(primitive action)과 복합 행위(composite behavior): 자율 에이전트(autonomous agent) 또는 능동 객체(active object)라고 불리우는 객체의 행위를 보면 행위는 주위 환경에 대응하여 매우 복잡하게 이루지는 것처럼 보인다. 그러나 실제 이런 복잡한 행위들도 살펴보면 매우 간단한 동작의 연속된 조합임을 알 수 있다. 사람 에이전트의 행위의 경우, 에이전트를 구성하는 머리객체, 몸객체, 팔객체, 다리객체의 기본적 행동에 의해 복잡한 행위-예를 들어 걸어가는 행위, 물건을 드는 행위, 문을 여는 행위, 그리고 이러한 복잡한 행위들의 복합적 조합에 의한 학교에 등교하는 행위, 여행을 떠나는 행위 등-의 자연스러운 표현이 가능하게 된다. 본 논문에서는 에이전트의 행위를 가장 기본적인 단위행동인 단위 행동과 이들의 복합적 조합으로 이루어지는 복합행위로 구분하여 복잡하고 동적이며 또한 개인에 따라 다양한 행위를 구현하기 위한 구조화가 가능하도록 하였다.

단위행동은 에이전트의 속성값의 변화를 가져오는 에이전트의 기본적 행위이다. 예를 들어 사람 에이전트가 팔을 들어올리거나 내리는 행동, 물건을 집는 행동, 발걸음을 옮기는 행동 등은 단위행동에 포함되어지며, 팔의 위치, 손에 물건이 들려졌는가에 대한 여부, 다리 객체의 위치와 사람 객체의 위치 등의 기본적인 속성값들에 변화를 가져오게 된다. 복합행위는 유기적으로 연관된 일련의 단위행동들, 또는 단위행동과 또 다른 복합행위의 연속된 조합에 의해 수행되어지는 추상적 개념의 행위이다. 여기서 추상적 개념의 행위의 의미는 그 자체로서는 에이전트가 어떠한 행위를 한다고 할 수 없지만 구성되어지는 다른 단위행동에 의해 에이전트의 현재상태가 변화되어지고 이러한 단위행동과 이들의 집합체인 또다른 복합행위의 동적인 일련된 과정이 모두 수행될 경우 복합행위도 수행된다는 의미이다. 예를들어 에이전트가 사물객체를 옮기는 행위 TakeTo($Object, $Space)의 경우는 다음의 계층구조를 통해 구조화될 수 있다. 그림의 내용은 주어진 바대로 자명하다.

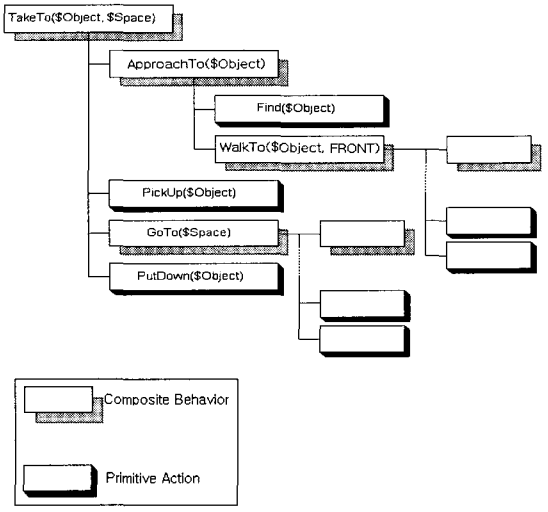


그림 8. 행위의 계층구조
Fig. 8. Behavior hierarchy.

본 논문에서는 행동에 관한 규칙은 아래와 같은 triplet으로 표현된다.

Action_name : (Preconditions, Procedural steps, Postconditions)

Preconditions은 행동을 실행하기 위한 초기조건으로서, 이 조건을 만족할 경우에만 행동이 실행되어지며 만족되어지지 않을 경우 에이전트는 다른 단위행동 또는 복합행위를 선행하여 이 조건을 만족하도록 동적으로 planning을 수행한다. Postconditions는 행동이 실행된 후 에이전트의 변화된 상태를 나타내며 이는 연속된 다음 행동의 Preconditions의 만족여부에 대한 판단 기준으로 이용되어진다. Preconditions와 Postconditions는 first-order predicate logic으로 표시된다. Procedural steps는 에이전트가 수행해야 할 기본적 작업의 순서를

표시하며 단위행동 또는 다른 복합행위를 포함할 수 있다. 예를들어 에이전트가 주위에 있는 물건을 들어올리는 행동 PickUp($Object)는 다음과 같이 정의되어질 수 있다.

```
PickUp($Object) :
    Preconditions : =($Agent.Hand, $Object)
                    ¬ATTACHED($Object,
                    $Agent.Hand)
    Procedural steps :
        Step 1. Find $Object.Object_ID.
        Step 2. Insert $Object.Object_ID in
        Attached_Object List of $Agent.
        Step 3. Attach $Object to
        $Agent.Hand.
    Postconditions :
                    ATTACHED($Object,
                    $Agent.Hand)
```

주어진 단위행동은 에이전트의 손객체가 집어 올리고자하는 물건객체 가까이 접근되어져있을 경우에 실행되어지며, 만약 객체가 에이전트로부터 멀리 떨어져 있을 경우 에이전트는 먼저 객체 주위로 이동하는 행위를 먼저 수행하여 Preconditions를 만족하도록 한다. 그림 8에 나타난 물건객체를 다른 장소객체로 이동시키는 복합행위 TakeTo($Object, $Space)는 다음과 같이 규정될 수 있다.

```
TakeTo($Object, $Space) :
    Preconditions : =($Agent.InSpace, $Object.
                    InSpace)
                    ¬IN($Object, $Space)
    Procedural steps :
        Step 1. ApproachTo($Object)
        Step 2. PickUp($Object)
        Step 3. GoTo($Space)
        Step 4. PutDown($Object)
    Postconditions :
                IN($Object, $Space)
```

위에서 보인 바와 같이 복합행위는 단위행동과 또 다른 복합행위로 이루어진 추상적 개념의 행위이므로 실제로 에이전트의 상태를 변화하는 Procedural steps를 가지는 것이 아니라 관련되어진 다른 복합행위 또는 단순행동들을 호출하는 형태로 이루어지는 점이 단위행동에 관한 정의와 차이점을 보인다.

복합행위의 구성요소의 변화: 복합행위의 구성은 에이전트나 다른 객체와의 상호작용을 통해 기본적 형태를 부여받는다고 하더라도, 환경과 자신의 상태에 따라 구성요소 중 일부분의 행위가 생략되어지거나, 수행이 불가능한 행위를 대체할 다른 복합행위를 동적으로 추가할 수 있는 판단구조를 필요로 하게 된다. 예를 들어 복합행위 GoTo($Space)의 경우 단순히 목적하는 공간객체가 근처에 있다면 걸어가는 복합행위를 필요로 하겠지만 거리에 따라 교통수단을 이용해야하는 부가적 복합행위를 요구할 수도 있게 된다. 복합행위의 구성요소의 변화는 크게 Step의 생략과 Step의 변환으로 나누어 볼 수 있다.

■ Step의 생략

하나의 복합행위가 가지는 각 Step은 또 다른 복합행위들 또는 단위행동들로 이루어져있으며 이러한 복합행위나 단위행동등은 각자 Precondtions와 Postconditions를 가지고 있다. 이 때 Postconditions는 해당 행위 또는 행동이 종료되어져야 할 조건을 나타내고 이는 에이전트가 요구되어진 행동이나 행위의 목적으로서 역할을 가진다. 그러나 에이전트의 상태가 실행하고자 하는 행동이나 행위의 Postconditions를 현재 만족하고 있다면 이는 에이전트가 요구되어진 행동이나 행위의 목적을 이미 달성한 것으로 판단되어질 수 있다. 즉, 행동이나 행위의 실행과정의 첫 단계에서 현재 에이전트의 상태와 Postcondtions를 비교하여 동일한 상태를 가질 경우 수행하고자 하는 행동이나 행위를 종료함과 동시에 생략하는 과정이 필요하다. 예를들어 에이전트가 TakeTo($Object, $Space)라는 복합행위를 수행할 경우에 있어서 요구되어진 객체가 이미 손에 들려진 상태라면 객체를 찾아서 들어올리는 행위는 불필요하게 된다. 따라서 TakeTo($Object, $Space)의 정의된 Steps 중에서 Step 1 과 Step 2는 생략되게 된다.

■ Step의 변환

하나의 복합행위를 에이전트가 실행하고자할 경우 환경과 에이전트 자신의 상태에 따라 좀 더 구체적인 복합행위로의 변환이 이루어질 수 있다. 이는 일부 복

합행위가 다른 복합행위들의 집합에 대한 추상적 개념이 될 수가 있기 때문이다. 예를 들어 '가다'라는 행위와 상응하는 복합행위 GoTo($Space)는 실제 걸어가는 행위, 뛰어가는 행위, 또는 탈것을 타고 가는 행위 등에 대한 집합적 개념의 복합행위이다. 따라서 에이전트는 GoTo($Space)의 실행단계에서 이동하고자 하는 공간 객체의 거리의 멀고 가까움, 교통수단의 존재, 에이전트에게 주어진 시간의 정도에 따라 실행하고자하는 복합행위는 좀더 구체적인 행위인 WalkTo($Space), RunTo($Space), TakeOn($Vehicle)∧TakeOffAt ($Vehicle, $Space) 등으로 변환되어질 수 있다. 이러한 복합행위의 변환을 이루기 위해서는 에이전트의 지식베이스에 포함되어진 복합행위의 규정시 집합적 개념의 복합행위일 경우 자신의 상태와 환경에 따른 구체적 복합행위를 판단하여 조건을 만족시키는 구체적 행위로 변환할 수 있는 구조를 포함시켜야 한다.

3. 객체의 특성과 에이전트와의 관계에 따른 행동양식의 분산

가상현실에서 이루어진 에이전트의 지식베이스 설계시 가장 문제시되는 부분 중의 하나는 모든 관련된 객체의 규칙과 특성에 관련된 지식을 전체적으로 하나의 에이전트 클래스에 구조화시키기 어렵다는 것이다. 이는 에이전트의 지식베이스에 저장되는 데이터양의 엄청난 증가와 더불어 실제 에이전트가 요구되어진 지식을 얻기 위한 추론에 지나치게 많은 작업시간이 소요되기 때문이다. 일례로 rule-based model[15]의 경우 동적인 환경에서의 모든 가능한 행위들의 규칙을 중앙제어 장치(Central Control Process)에서 저장하고 처리한다. 따라서 이러한 경우에 있어서 행위의 각 Step마다 매우 많은 규칙의 결정트리(Decision tree)를 탐색하여야 하므로 동적 환경의 미세한 변화라 하더라도 결정트리에 큰 변화를 필요로 하게되고, 또한 행위의 재정렬이 힘들게 된다. 예를들어, 문 객체와 사람 에이전트와의 관계를 살펴보자. 문 객체의 종류는 미닫이문(sliding door), 여닫이문(hinged door) 등으로 다양할 뿐 아니라 문 객체를 이루는 구성요소에 따라 에이전트가 문 객체를 여는 행위 Open($Door)는 다양하게 규정되어야 된다. 따라서 관계된 행동양식을 모두 에이전트의 지식베이스에 저장하는 것은 지난하고, 한편 다른 특성과 규칙을 가지는 문 객체의 추가 시 지식 베이스의 수정이 필요하게 된다.

이러한 문제를 극복하고 에이전트의 행위의 효과적 정의를 위해 본 논문에서는 개별 에이전트는 관계되어지는 종류의 객체에 대한 학습된 정도 그리고 실제로 에이전트가 객체에 대한 인식 정도에 관한 지식만을 보유하게 되고, 동일한 종류의 객체 일반에 적용 가능한 복합행위들은 그들의 이름만을 기억하도록 설계한다. 그리고 복합행위에 대한 규칙은 구체적 시공간 상황에서 객체와의 상호 메시지 교환을 통해 동적으로 해당 객체로부터 받아온다. 즉 에이전트의 지식베이스에는 상황에 등장하는 객체에 대한 학습이나 인식 여부 그리고 객체에 관련된 행위의 종류만을 저장하고, 등장 객체들의 지식베이스에 관련된 복합 행위들에 관한 production rule들을 저장하여 이들을 에이전트가 요구할 경우 메시지로써 에이전트에게 전달한다. 이 때 메시지를 전달받은 에이전트는 자신의 학습 및 인식 정도에 따라 production rule의 Step들을 재조정하여 이를 실행에 옮기게 된다.

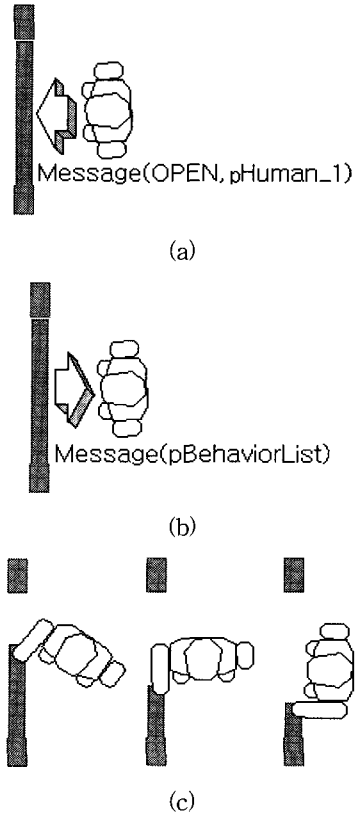


그림 9. 문 객체와 사람 에이전트간의 상호작용
Fig. 9. Interaction between human agent and door object.

그림 9는 사람 에이전트가 복합행위 Open($Door)를 실행하기 위하여 관계되어지는 문 객체와의 상호작용을 통하여 복합행위에 대한 production rule의 Procedural Steps를 취득하여 이를 실행하는 것을 보여준다.

그림 9의 (a)에서 에이전트는 현재 자신이 수행하고자 하는 행위에 대한 구체적인 production rule이 모두 자신의 지식베이스에 저장되어 있지 않고 다른 객체에 분산되어 있는지를 파악한다. 분산되어 있는 경우 에이전트는 해당 객체에 메시지 전달을 통해 행위에 대한 지식을 요구한다. 구체적으로 에이전트가 Open(Door_1)이라는 복합행위를 수행하기 위해 Door_1에 다음의 메시지를 전달한다.

Message(OPEN, pHuman_1)

첫 번째 파라미터는 자신이 기억하는 행위의 이름이며 두 번째 파라미터는 복합행위에 대한 행위양식을 요구한 에이전트 자신의 접근 포인터이다. 여기서 두 번째 파라미터는 자신의 Object_ID로서도 대체 가능하다. 그림 9의 (b)에서 복합행위에 대한 세부적인 행위양식을 요구받은 객체는 자신의 지식베이스를 탐색하여 행위양식의 종류와 그러한 행위양식을 요구한 에이전트의 현재상태 그리고 자신의 상태 등을 고려하여 필요한 행위양식을 리스트 형태로 작성하여 이를 요구한 에이전트에게 역시 메시지형태로써 되돌려준다. 그림 9 의 (c)에서 에이전트는 전달받은 행위 리스트와 자신의 상황에 근거하여 자신의 행위를 동적으로 planning하여 이를 실행한다.

이와 같이 객체의 특성과 에이전트와의 관계에 따른 규칙의 분산과 객체와 에이전트 사이의 메시지 전달을 통한 인식과정은 실제 실생활에서 사람이 인식기관을 통해 객체를 인식하여 자신의 지식구조에 따라 행동양식을 결정하는 것과 유사한 방법으로 가상현실을 이용한 몰입적 교육시스템에서 효율적인 방법론이 된다. 또한 개별 에이전트마다 자신의 지식베이스에 거의 동일한 행위 양식들을 반복적으로 가지는 비효율을 최소화할 수 있다.

4. 에이전트의 행위와 관련된 지식 구조

에이전트의 행위를 크게 두 가지로 나누어 앞 절에서 기술하였다. 에이전트가 복합행위와 단위행동을 자신의 상태와 주위 환경에 적합하게 실행하기 위해서는 적절한 지식 구조들을 필요로 하게 된다. 본 절에서는 필요한 지식 구조를 정의하고 이를 이용한 동적 planning과정을 논한다. 에이전트가 행위와 관련되어 가지는 지식 구조는 명령 리스트(List of Commands), 복합행위 리스트(List of Behaviors) 그리고 단위행동 리스트(List of Actions)의 세 가지로 나누어진다. 명령 리스트는 현재 에이전트가 부여받은 가장 상위의 복합행위를 저장하는 공간이다. 예를들어, "John is going to school."의 경우 이름(Object_Name)이 'John'인 에이전트의 명령리스트에는 복합행위 GoTo($School)가 저장되어 실행되어지게 된다. 명령 리스트에 포함되어지는 복합행위는 현재 에이전트의 행위를 종합적으로 표현하는 것으로서 외부로부터 주어진 시점에서 에이전트가 행하고 있는 전체적 행동상황에 대한 질의를 받을 경우 이에 응답이 될 수 있는 지식구조이다. 복합행위 리스트는 복합행위로 분류되어진 행동양식을 저장하는 구조로서 구체적으로 명령 리스트에 포함되어진 복합행위의 Procedural steps를 저장한다. 에이전트는 저장되어진 리스트의 첫 번째 요소부터 순차적으로 행위를 시작하게 되며, 수행되어지는 복합행위의 Procedural steps가 또 다른 복합행위를 포함하게 될 경우에는 복합행위 리스트의 앞부분에 이를 스택구조의 순서에 따라 동적으로 삽입하게 된다. 마지막으로 단위행동 리스트는 실제적으로 에이전트가 수행하게 되는 단위행동을 저장하기 위한 구조이다. 복합행위 리스트에서 수행하고자하는 복합행위의 Procedural steps에 단위행동이 포함될 경우 이를 단위행동 리스트에 옮겨지고, 단위행동 리스트에 포함된 이러한 단위행동만을 에이전트는 실제적으로 수행하게 된다. 수행 후 에이전트는 완료된 단위행동을 리스트에서 제거시킨다. 이러한 지식구조를 도식적으로 표현하면 그림 10과 같다.

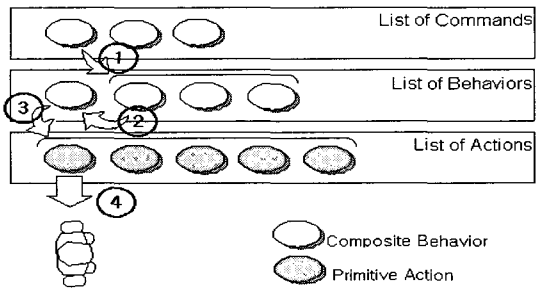


그림 10. 동적 환경에서의 행위 구현을 위한 에이전트의 지식구조

Fig. 10. Agent's knowledge structure for behaviors in dynamic environment.

그림 10에서 ① 명령 리스트에 포함되어진 복합행위의 Preconditions가 현재 에이전트의 상태를 만족할 경우, Procedural steps에 포함되어진 복합행위를 복합행위 리스트에 포함시킨다. ② 복합행위 리스트의 첫 번째 요소의 실행 시 Procedural steps에 또 다른 복합행위가 포함될 경우, 이들을 리스트의 첫 요소에 삽입시킨다. 결과적으로 원래의 복합행위는 리스트 상에서 후위로 밀려나며, Procedural steps상의 다른 모든 행위가 먼저 실행되어 실제 행위가 두 번 연속으로 실행되는 것으로 될 수 있으나, 뒤로 밀려난 복합행위는 실행 시 에이전트의 상태가 행위의 Postconditions를 만족하게 되므로 생략되게 된다. ③ 복합행위의 Procedural steps에 포함되어지는 단위행동을 단위행동 리스트에 삽입하여 실제 에이전트가 최초 명령받은 행위를 단계별로 실행할 수 있도록 한다. ④ 에이전트는 단위행동 리스트의 첫 번째 요소부터 실행에 옮겨 실제 에이전트의 상태 변화와 이에 따른 전체적 환경의 변화를 가져오게 된다.

제시되어진 세 가지의 리스트 형태의 지식 구조와 복합행위, 단위행동의 규칙, 그리고 다른 객체와의 관계를 고려한 행위양식을 분산을 이용할 경우, 에이전트는 planning을 하는 과정에 있어서 동적으로 환경에 반응하여 필요한 조건 검색, 조건 만족을 위한 다음 행위 결정, 그리고 수행이 뒤따르게 되므로, 행동의 각 단계마다 가능성을 가지는 모든 행위를 찾기 위해 많은 규칙의 결정트리를 탐색해야 하는 Rule-based model[15]과 대비하여 필요한 지식을 위한 탐색 시간이 줄어들어 효율적으로 행위를 구현할 수 있게 된다.

Ⅳ. 구현 및 실험 결과

본 실험에서는 사람 에이전트가 복합적인 공간을 옮겨다니는 상황을 Windows 운영체제 환경에서 Visual C++ 6.0과 MFC를 사용하여 시뮬레이션하였다. 그림 11은 시뮬레이션 되어지는 시공간 상황의 초기화면을 보여 준다.

시뮬레이션을 위한 윈도우는 그림 11에서와 같이 크게 두 부분으로 나누어 볼 수 있다. 윈도우의 좌측의 화면(①)은 설정된 환경에서의 에이전트와 객체의 이미지를 사용자에게 전달하기 위한 이미지뷰(Image View)로서 전체적인 시공간 상황을 사용자에게 제공한다. 그리고 윈도우 우측의 화면(②)은 사용자의 요구에 의해 선택되어진 객체 또는 에이전트의 현재 상태를 화면에 표시하며, 에이전트의 경우에는 현재 실행 가능한 행위의 목록을 리스트 박스에 표시하여 이에 대한 요구를 받아 에이전트에게 전달하는 역할을 담당하게 된다. 본 실험에서 생성되어지는 객체와 에이전트는 객체지향적인 방법에 의해 생성되어지고 메시지를 통해 정보를 교환한다[17,18]. 먼저 상황의 배경이 되는 공간 객체의 생성과 이에 관계되어지는 객체들에 대해 알아본다.

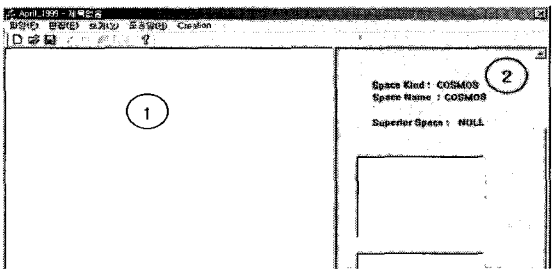


그림 11. 초기 윈도우
Fig. 11. Initial window.

초기 윈도우가 생성됨과 동시에 가장 최상위의 공간객체인 우주 공간객체(COSMOS)가 그림 11의 이미지 뷰를 포함하는 영역의 크기로 생성되어진다. 따라서 이후에 생성되어지는 모든 공간객체는 그 최상위객체로 공간객체 COSMOS를 가지게 되며 따라서 공간객체 COSMOS가 가지는 규칙과 특성을 상속받게 된다.

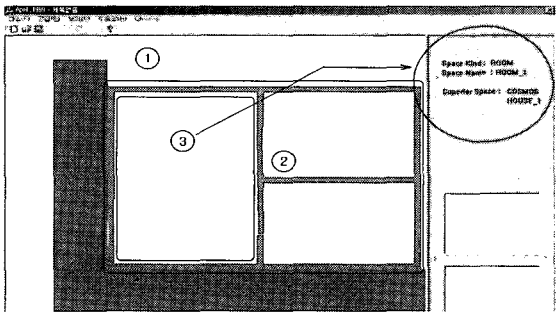


그림 12. 기본적 집 객체에서의 공간 객체의 생성
Fig. 12. Creation of the House object as a basic Space object.

그림 12는 사용자의 요구에 따라 전체적인 상황의 기본적 환경을 제공하는 공간객체로서 집(House) 객체를 생성하고 초기화시킨 그림이다. 실험에서 생성되어지는 공간객체의 기본적 구성은 앞에서 설명되어진 그림4와 같은 공간계층구조를 가지도록 설계되었다. 사용

자의 마우스 입력에 따라 지정되어진 공간객체의 정보를 상태뷰에 표시하게 된다. 그림 12는 사용자에 의해 공간객체 Room_1의 정보를 보여준다. 공간객체 Room_1은 {COSMOS, House_1}을 상위객체로 가지게 됨을 상태뷰를 통해 확인할 수 있다. 이를 이용하여 사용자는 입력하고자 하는 공간객체의 장이 영향을 받게 되는 공간을 확인할 수 있다. 일반적 집객체의 생성시 문객체는 집객체의 기본 요소로서 자동적으로 생성되어 공간객체 House_1, Room_1, Room_2, Room_3의 구성요소로 포함되어진다. 문객체의 생성 시 자신의 주변을 탐색하여 자신이 어떤 공간 객체들을 서로 연결하게 되는지 동적으로 확인하여 확인된 공간객체에 메시지를 통하여 인접공간객체(Adjacent Space) 정보의 확보가 가능하도록 한다.

상황에 등장하여 에이전트로서의 역할을 가지는 사람객체는 객체지향기술을 바탕으로 하여 그림13에서 보여지는 바와 같이 사람객체를 구성하는 머리객체(CHead), 몸객체(CBody), 팔객체(CArm), 다리객체(CFoot)를 기본 구성 객체로 하여 이를 전체적으로 관리하는 사람객체(CHuman)는 복합객체로 설정한다[19]. 사람객체의 생성 시 동적인 환경의 설정에 의한 다양한 상황을 표현하기 위해 사람객체의 생성위치와 방향은 사용자의 제어에 의해 임의로 설정되도록 한다. 사람객체는 자신이 속한 환경을 인식하기 위해 계속해서 존재하는 공간영역을 확인한다.

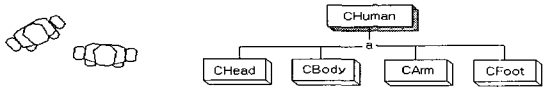


그림 13. 사람객체의 클래스 계층구조
Fig. 13. Human icons and class hierarchy.

그림 14는 사람객체에 대한 사용자의 제어화면을 보여주는 그림이다. 사용자가 왼쪽 마우스로서 사람객체를 선택한 경우, 해당 사람객체에 대한 정보를 상태뷰에 표시한다. 이는 사람객체의 Object_ID, 위치, 방향, 그리고 장의 영향을 받고 있는 공간객체의 리스트를 보여준다. 선택된 사람객체가 바뀌어질 때마다 해당 객체에 대한 정보를 상태뷰에 나타낸다. 그리고 상태뷰의 아래쪽에 나타나는 리스트박스에는 현 상황에서 사람객체가 수행 가능한 복합행위의 이름을 나열하였다. 사용자가 리스트박스 내의 하나의 복합행위를 더블클릭할 경우 해당되어지는 사람객체는 그에 대한 행위를 시작하게 된다.

사람 에이전트의 행위가 시작되면 상태뷰의 두 번째 리스트박스에는 현재 사람 에이전트가 주어진 작업을 수행하기 위해 계획되어진 행위의 일반적인 순서가 사용자에게 보여진다. 이는 사람 에이전트가 행하는 작업, 현재 자신의 상태에 따라 실행 시 복합행위의 생략, 변환 등의 과정이 반복되어지면서 나타난다.

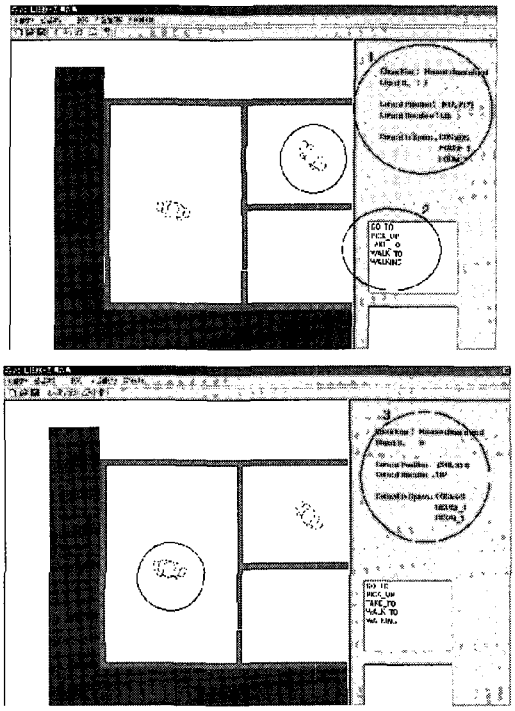


그림 14. 사람객체에 대한 사용자 제어화면
Fig. 14. User's control screen over Human object.

그림 15 (a)는 사람 에이전트가 공간객체 Room_3에서 공간객체 Room_1으로 이동하는 복합행위의 초기장면을 나타내며, 초기에 자신이 속한 공간객체와 이동하고자하는 공간객체가 다르므로 이에 대한 planning 과정을 통해 문객체 Door_3을 통하여 이동이 가능함을 판단하고 그에 기초하여 동적으로 이동행위를 수행하는 그림이다. (b)의 장면에서는 사람 에이전트 자신의 지식 베이스에 포함되어있지 않은 Open($Door)에 대한 행위규칙을 문객체와의 상호작용을 통해 습득하는 과정을 보여준다. (c)와 (d)는 이러한 상호작용에 의해 인식하게된 Open(Door_3), Go_Through(Door_3)의 행위를 실제 자신의 행위리스트에 포함되어짐을 보여주는

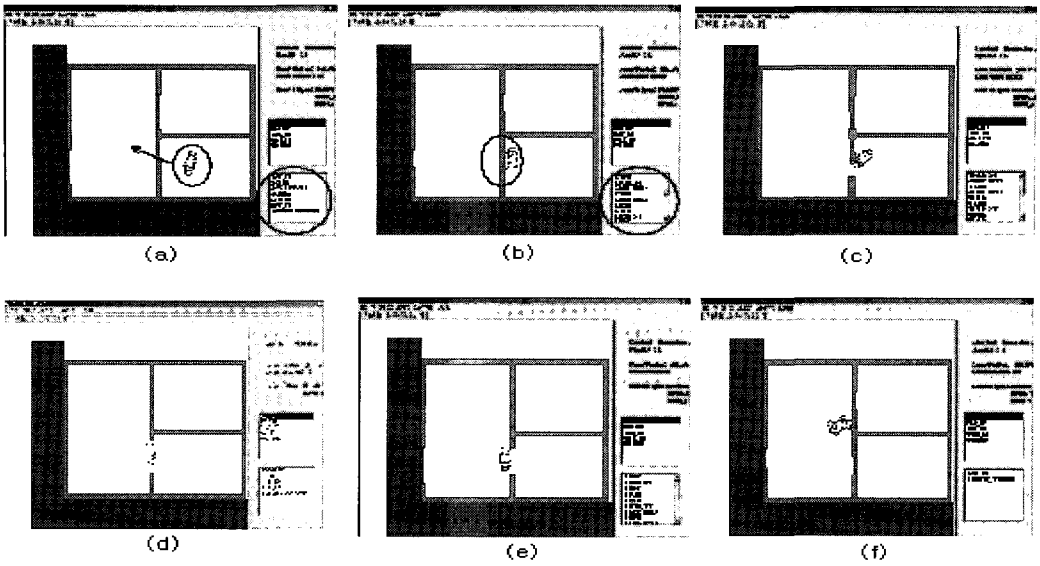


그림 15. 사람 에이전트의 공간이동
Fig. 15. An overall space navigation of human agent.

과정이다. (e)의 화면은 Close(Door_3)의 행위에 대한 화면이며, (f)는 마지막으로 최초 사람 에이전트가 이동하고자 하던 공간객체 Room_1내의 목표 지점으로 이동하는 화면이다. 상황의 흐름에 따라 초기 planning에 의해 사람 에이전트의 행위리스트에 포함되어진 복합행위나 단위행동들은 단계적 수행을 거쳐 전체적 행위를 구성하게 된다.

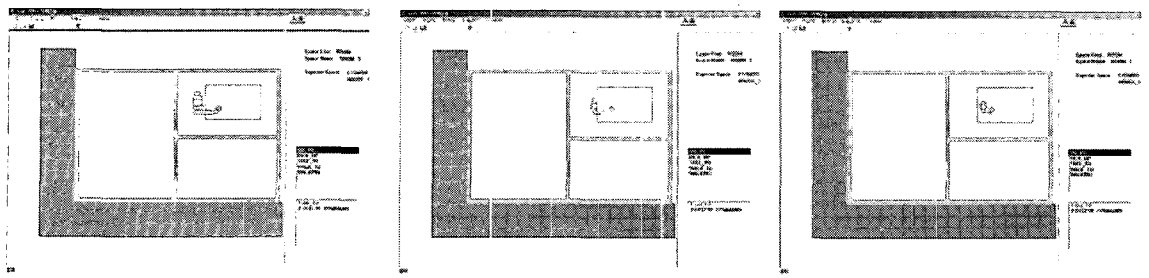
V. 실험 결과에 대한 검토

본 연구의 궁극 목표는 에이전트가 변화하는 상황에서 대처하여 자율적으로 판단하여 행동할 때 필요한 지식과 상황을 모델링하는데 있다. 구체적으로 동적인 환경에 대처하는 에이전트의 지능이나 능력에 따라 생길 수 있는 여러 가지 다양한 상황을 체계적으로 수용하는 모델의 개발에 초점을 맞추었다. 본 모델에 적용 가능한 수많은 종류의 상황들 중에서 비교적 명백한 평가가 가능하다고 보아지는 공간 이동을 기초로 하는 예제 상황으로 선택하였다. 다른 공간 이동에 관한 모델들 중 앞서 언급한 바와 같이 가장 관련성이 많은 OGH (Object Geometric Hierarchy) 모델과의 차이를 중심으로 항목별로 기술한다. 이들 모델과의 차이점은 주로 정성적인 면을 중점적으로 항목별로 언급한다.

첫째, 두 지점 사이의 공간 이동시 OGHM의 경우 단순히 이동 경로만을 제시할 수 있는 반면, 본 접근 방법은 경로 상에서 발생할 수 있는 문제점들을 해결하기 위한 행동을 구체적으로 묘사할 수 있다. 예를 들어 그림 15 (c) - (e)에 서 보여주는 바와 같이 이동 경로 상에 문이 있다면 문을 여는 방법 (절차)을 규정할 수 있다.

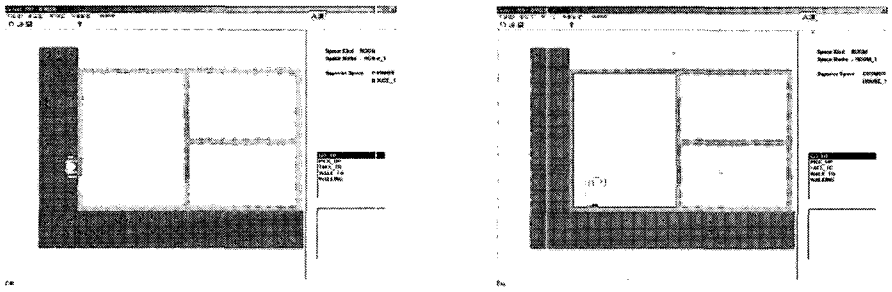
둘째, OGHM과 달리 공간객체와 사물객체를 구별함에 따라 유도공간이라는 개념이 추가됨으로써 사물 객체 주변에서 생길 수 있는 상황을 구체적으로 묘사할 수 있다. 예를 들어, 책상 객체가 추가되고 그 위에 있는 컵을 잡는 경우, 그림 16 (a)에서 보는 바와 같이 어른 에이전트는 책상 앞에서 (더 이상 공간 이동 없이) 손을 뻗쳐 잡게되고 그림 16 (b)와 (c)에서 보는 바와 같이 팔이 짧은 아이 에이전트는 책상 위를 올라가서 잡으려 하는 상황을 표현할 수 있다. OGHM의 경우라면 에이전트의 속성과 관계없이 Move(Study, Desk) > Move(Desk, Cup_2)로 표현함으로써 단순히 방 객체에서 책상 객체로 이동하면 된다고 규정하는데 그칠 것이다[16].

셋째, 에이전트가 상황의 변화에 대처하는 행위를 동적으로 표현할 수 있다. 예를 들어 그림 17 (a)에서 보는 바와 같이 에이전트가 문을 열고 보니 실내가 어둡다는 사실을 발견했을 때 그림 17 (b)에서와 같이 불을 켠다든지 하는 수단을 강구하는 행위를 표현할 수 있다.



(a) (b) (c)

그림 16. 동일한 행위에 대한 에이전트의 특성에 따른 상이한 대처의 예
Fig. 16. Difference in execution of the same action according to agents' characteristics.



(a) (b)

그림 17. 동적 환경 변화에 대한 에이전트의 대처의 예
Fig. 17. Agent's example reaction to dynamic environmental change.

넷째, OGHM에서는 동일한 상위공간 객체 (A) 에 속하는 두 하위공간 객체 (B to C) 사이를 이동할 경우에도 B에서 상위 객체 (A)로 이동한 후 다시 C로 이동하는 경로[16]를 택하는 반면, B에서 C로 바로 (A를 거치지 않고) 가는 것이 실제 사람 객체의 논리에 맞다. 구체적으로 본 논문에서는 planning 과정에서 상충되지 않는 다른 공간으로의 이동으로 대체되나 OGHM은 본 논문의 규칙 1과 같은 수준의 planning 에 그치게 된다.

다섯째, 에이전트의 행위들을 기술하는 개별 절차들을 각 에이전트의 지식 베이스에 위치시키는 경우 에이전트의 숫자에 비례하여 증가하게 될 것이다. 반면 개별 에이전트의 특성과 관계없이 거의 일정한 절차 부분을 에이전트가 아닌 대상 객체에 위치시킴으로써 에이전트의 수에 관계없이 일정하게 된다. 예를 들어 문의 종류마다 하나의 절차를 각 에이전트마다 정의하기 보다 문객체에다 일정한 절차를 저장하고 각 에이전트에는 개별 에이전트에 특수한 부분만 추상화시켜 정의함으로써 문을 여는 행위에 대한 지식 저장 및 search 효율을 증가시킬 수 있을 것이다.

VI. 결론 및 향후 연구 방향

본 논문에서는 가상 환경에서의 교육시스템에서 필요로 하는 자율 에이전트의 행위의 구현을 위한 방법론으로, 장이론을 이용한 공간객체의 설정과 에이전트의 행위와의 연계, 단위행동(Primitive Action)과 복합행위(Composite Behavior)의 조합으로 가능한 다양한 행위표현의 방법, 에이전트와 객체의 상호관계에 의한 규칙의 분산 등을 개발하였다. 이러한 방법론들에 기초하여 다양한 상황의 구축과 이러한 환경에 동적으로 반응하는 자율 에이전트의 구현을 통해 몰입적 교육시스템을 위한 실질적 기초인 가상상황을 제공할 수 있음을 보였다. 또한 설정되어진 구조가 실제 물리적 세계에서 학습자가 인식하는 방법과 유사하므로 학습의 기대를 증대시킬 수 있다.

앞으로의 연구방향으로 공간객체뿐만 아니라 자율 에이전트에도 장이론을 확장시켜 심리적 장에 의한 행

위 유발에 관한 연구, 에이전트의 지식베이스에서 장기 기억(long-term memory)와 단기 기억(short-term memory)의 구별에 따른 행위의 진보, 퇴행, 오류에 관한 연구 등이 함께 이루어지면 좀 더 복잡하고 동적인 환경과 이에 대한 에이전트의 적절한 행동을 구현할 수 있을 것이다.

참 고 문 헌

[1] Yukihiro Matsubara, Seiji Toihara, Yuichiro Tsukinari, Mitsuo Nagamachi, "Virtual Learning Environment for Discovery Learning and its Application on Operator Training," IEICE Transactions on Information & Systems, V.E80-D N.2, Feburary 1997.

[2] 양진모, 심임섭, "Multimedia Data를 이용한 Intelligent Tutoring System의 Tutoring Module 설계," 1997년도 한국정보과학회 가을 학술발표논문집 Vol.24 No.2, pp.43~46

[3] Charles E. Hughes, J. Michael Moshell, "Shared Virtual Worlds for Education : The ExploreNet Experiment," ACM Multimedia 1997.

[4] W. Lewis Johnson, Jeff Rickel, Randy Stiles, Allen Munro, "Integrating Pedagogical Agents into Virtual Environments," Proceedings of the International Conference on Computer and Education, 1995.

[5] Stuart J. Russell and Peter Norvig, "Artificial Intelligence : A Modern Approach," Prentice Hall Inc. pp.31~52, 1995.

[6] Pattie Maes, "Artificial Life meets Entertainment: Lifelike Autonomous Agents," Communications of the ACM, Vol.38, No. 11, November 1995.

[7] Pattie Maes, "Modelling Adaptive Autonomous Agents," Artificial Life Journal, edited by C. Langton, Vol.1, No.1&2, pp. 135~162, MIT Press, 1994.

[8] Kurt Lewin, "Behavior and development as a function of the total situation," Field Theory of Social Science, pp.285~354, 1963.

[9] Toshio Fukuda, Naoyuki Kubota, "Adaptation, Learning and Evolution," 1998 Second International Conference on Knowledge-Based Intelligence Electronic Systems, 21~23 April 1998.

[10] K. Sata, N.Baba, "A Research Concerning a Concept Generation and an Action of an Agent," 1998 Second International Conference on Knowledge-Based Intelligent Electronic Systems.

[11] 배경표, "동적 환경에서의 능동 에이전트간 상호작용의 힘의 장에 기반한 formalism," 경북대학교 석사학위논문, 1998

[12] Donna Coco, "Creating Intelligent Creatures," Computer Graphics World, July 1997.

[13] 주우석, 최성운, 박경희, 이희승, "가상현실을 위한 객체 연결 모델," '96 한국정보처리학회 논문지 제3권 제1호, pp. 95~106, 1996

[14] Murray, J. & Malone, S., "The Structures of Advanced Multimedia Learning Environments: Reconfiguring Space, Time, Story, and Text," ITS '92, Montreal, Canada, June 1992.

[15] Hanqiu Sun, "A Relation-Based Model for Animating Adaptive Behavior in Dynamic Environments," IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans, Vol.27, No.2, March 1997.

[16] Stanley Y. P. Chien, Lucy Q. Xue, and Mathew Palakal, "Task Planning for a Mobile Robot in an Indoor Environment Using Object-Oriented Domain Information," IEEE Transactions on Systems, Man, and Cybernetics - Part B: Cybernetics. Vol.27 No.6, December 1997, pp.1007~1016.

[17] Bogdan Czejdo, Christoph F. Eick, Malcolm Talyer, "Integrating Sets, Rules, and Data in an Object-Oriented Environment," IEEE Expert, February, 1993. pp.59~66.

[18] Roger Y. Lee, "Object-Oriented Software Engineering: An Object Modeling Technique," Proceedings of the Computer Science Group - The Association of Management(AoM) 13th Annual International Conference, 1995.

[19] V. Puig and C. Oussalah, "Constraints and Composite Objects," Proceedings of the 7th International Conference, DEXA '96, September 1996.

저 자 소 개

朴 亨 根(正會員)

1972년 10월 15일생. 1998 년 2월 경북대학교 전자공학과 학사. 2000 년 2월 경북대학교 전자공학과 석사. 2000 년 1월~ 현재 LG전자(주) 정보통신시스템 사업본부 생산기술 연구소 S/W기술실 주임연구원. 주 관심분야는 지능망, BISUP, DBMS, 전문가 시스템

朴 宗 熹(正會員) 第 32 券 B 編 第 12 號 參照