제23회 한국정보처리학회 춘계학술발표대회 논문집 제12권 제1호 (2005. 5)

실시간 반응형 로봇 행위 지정을 위한 에이전트 언어

곽별샘, 변무홍, 이재호
서울시립대학교 전자전기컴퓨터공학부
e-mail : semix2@naver.com, bwikbs@ece.uos.ac.kr, jaeho@uos.ac.kr

An Agent Language for Real-Time Reactive Robotic Behavior Specification

Byul Saim Kwak, Moo Hong Byun, Jaeho Lee
Dept. of Electrical and Computer Engineering, University of Seoul

요 약

본 논문에서는 실시간 반응형 로봇의 행위를 지정하기에 적합한 에이전트 언어를 소개한다. 기존의 BDI 기반의 에이전트 언어를 기반으로 실시간 반응형 로봇의 행위를 지정하는데 적합하도록 개발한 VivAce 에이전트 구조에 대해서 설명하고 이를 이용한 간단한 시뮬레이션을 수행하였다. 또한 VivAce 가 기존의 BDI 에이전트 언어에 비해서 가지는 새로운 특징인 자바 네이티브 언어 지원, 쓰레드 기반의 계획 실행, 다양한 인터페이스를 소개한다.

1. 서론

주변 환경을 인지하고 그 환경에 합리적인 행동을 결정하며 실제로 그 결정을 실행하는 단위로써 에이전트를 사용하자는 시도는 이미 어느 정도 문제를 해결하는 도구로서 유용성이 검증되어 왔다. 하지만 에이전트가 실제로 어떤 구조로 움직여야 하는 지에 관한 연구는 아직도 진행형이라고 할 수 있다. 초기에 규칙기반 에이전트 시스템에서 시작하여 행동기반 에이전트 시스템이 나왔고 계획기반 에이전트 시스템등 많은 시도가 있어 왔으나 각 시스템의 장단점이 분명하기 때문에 나중에 나온 시스템이 전에 나온 시스템을 대체한다라기 보다는 사용되는 용도에 적합한 접근법을 선택하는 것이 일반적이라 할 수 있다. 본 논문에서는 기존의 BDI 아키텍쳐 기반으로 만들어진 에이전트 시스템인 JAM 이 가지는 단점에 대해서 개선하고 좀더 빠르고 쉽게 사용하기 위해서 VivAce 라는 새로운 에이전트 시스템을 개발하였다. VivAce 는 자바 네이티브 언어를 사용해서 개발자들이 좀더 직관적으로 에이전트 시스템을 개발할 수 있는 환경을 제공한다. 또한 쓰레드 기반의 계획의 실행을 보장함으로써 개발된 시스템의 속도향상에 꾀할 수 있다.

본 논문에서는 환경의 변화를 감지하고, 감지된 변화를 내부 정보로서 표현하며 내부 정보에 근거한 판단을 통해 행동하는 BDI 아키텍쳐 기반의 에이전트 시스템인 VivAce 에이전트 시스템에 대해 설명한다. VivAce 에이전트 시스템은 환경 정보를 수집하고 적절히 가공하는 감지기(Sensor)와 가공된 정보를 표현하는 실세게 모델(WorldModel), 그리고 그것에 근거하여 시스템의 목표(Goal)를 성취할 수 있는 실행 계획(Plan)을 판단하고 선택하는 판단기(Reasoner), 실행중인 계획에 대해 그것이 수행할 수 있는 상황인가를 감시하는 상황 감시기 (ContextMonitor)로 구성되어 있다. 이러한 VivAce 에이전트 시스템을 이용하여 서비스 로봇을 구현하고 그것이 다양한 환경의 변화 속에서 시스템의 전체 목표를 위해 적절한 반응을 하며 수행하는 모습을 간단한 시뮬레이션을 통해 확인한다.

2. VivAce 에이전트

VivAce 에이전트는 BDI 아키텍쳐에 기반한 에이전트 구조를 갖는다. 그림 1 은 VivAce 에이전트 시스템의 구조를 보여준다.

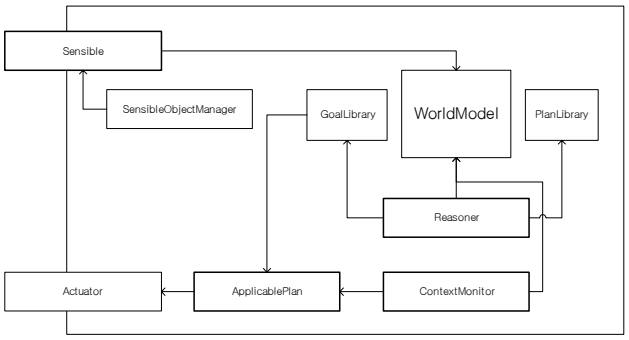

그림 1: VivAce Agent System

VivAce 에이전트는 크게 외부 환경을 감지하고 감지된 정보에 대해 가공된 정보를 제공하는 감지기(Sensor), 환경으로부터 감지된 정보와 시스템 내부에서 사용되는 정보를 포함하는 실세계모델(WorldModel), 에이전트의 목적을 포함하는 목적 라이브러리(GoalLibrary), 목적을 달성하기 위한 실행 요소들을 포함하는 계획 라이브러리(PlanLibrary), 목적을 수행할 수 있는 적절한 계획을 찾아 수행시키는 판단기(Reasoner), 실행중인 계획을 감시하고 계획이 요구하는 상황에 부합하지 않으면 실행중인 계획을 중지시키는 상황 감시기(ContextMonitor)로 구성된다.

에이전트가 수행해야할 목적은 목적 라이브러리(GoalLibrary)에 적재된다. 최초 에이전트의 목적은 사용자에 의해 적재되지만 수행중인 에이전트가 특수한 상황으로부터 그것을 해결하기 위한 목적을 스스로 적재할 수도 있다.

계획 라이브러리(PlanLibrary)는 앞서 말한 에이전트의 목적을 성취할 수 있는 계획(Plan)들의 집합이다. 각각의 계획은 특정 목적을 성취할 수 있는 행동과 그것을 실행하기위해 선행되어야할 환경 조건, 수행중에 계속 유지되어야 할 환경 조건, 계획이 실패할 경우 취해야 할 행동, 실행 계획이 갖는 효율성 등이 서술된다. 선행 조건과 유지 조건에 대해 좀 더 설명하자면, 예를 들어 컵을 A 위치에서 B 위치로 이동하는 실행 계획이 있을 때 'A 위치에 컵이 있고 그것을 집을 수 있다' 가 선행 조건이다,. 선행 조건을 만족하면 이 계획을 실행시켜 컵을 이동시킬 것이다. 이 때 유지 조건은 '컵을 들고 있고 B 위치에 컵을 놓을 수 있는 공간이 있다' 가 될 것이다. 이 계획이 수행 중일 때 만약 컵을 떨어뜨리거나 B 위치에 다른 것이 놓여있어서 컵을 놓을 수 없는 상황이라면 상황 감시기(ContextMonitor)에 의해 계획은 적절한 실패시 행동을 취하고 종료되며 판단기(Reasoner)에 의해 다른 적절한 계획이 선택될 것이다. 또는 실패시 행동이 현재 상황을 극복하기 위한 목적을 생성했다면 그것을 위한 계획을 수행시켜 이 상황을 극복할 것이다. 상황이 극복되면 실패했던 실행 계획이 다시 선택되어 목적을 수행할 수 있을 것이다.

감지기(Sensor)는 에이전트가 속한 환경의 특정 부분을 주기적으로 감지하고 적절한 환경 정보에 대해 가공하여 에이전트의 실세계모델(WorldModel)에 정보를 제공하거나 기존의 정보를 갱신한다. VivAce 에이전트는 다수의 감지기를 연결할 수 있기 때문에 각각의 감지기는 특수한 목적에 맞게 제작될 수 있다. 그림 2 는 환경을 감지하고 적절한 정보에 대해 그것을 가공하여 실세계모델에 입력하는 모습을 나타낸다.

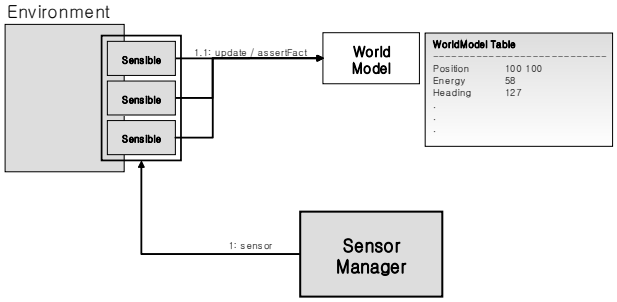

그림 2: 감지기에 의한 환경 지각

판단기(Reasoner)는 목적 라이브러리로부터 에이전트가 수행해야 할 목적을 검색하고, 그것을 성취할 수 있는 적절한 실행 계획을 계획 라이브러리로부터 찾아 실행시킨다. 각각의 목적에 대해 판단기는 일차적으로 그것을 성취할 수 있는 실행 계획들을 검색하고, 검색된 실행 계획들 가운데 선행 조건을 만족하는 것들을 추려낸다. 추려낸 계획이 다수일 경우 목적에 대한 계획의 효율성을 서술된 정보로부터 추론하고 효율성이 가장 높은 계획을 수행시킨다. 그림 3 은 이러한 과정을 도식화 한 것이다.

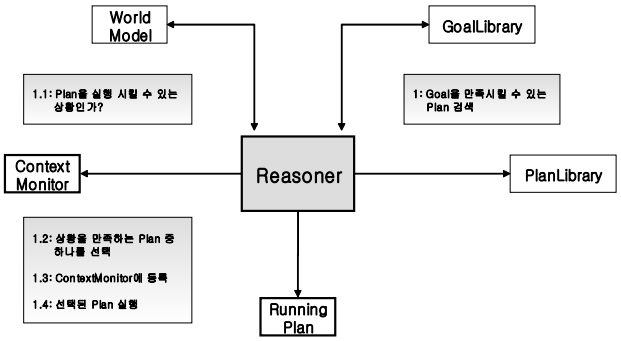

그림 3: 판단기에 의한 실행 계획 선택 및 수행

실행중인 계획에 대해서는 상황 감시기(ContextMonitor)가 주기적으로 실행중인 계획이 명시한 유지 조건에 대해 감시한다. 실행중인 계획이 명시한 유지 조건을 현재 실세계모델이 만족하지 못하면 상황 감시기는 해당 계획을 종료시킨다 (그림 4). 이 때 계획은 바로 종료되는 것이 아니라 그것에 정의된 실패시 행동 절차를 수행하고 종료된다. 이것은 실행중인 계획을 안전하게 종료시키기 위한 절차이다. 계획의 실행을 위해 확보한 자원을 안전하게 반환하거나 처음 상태로 되돌리는 등의 행동이 기술된다. 또는 실패의 원인을 분석하고 실패 원인을 제거시키기 위한 새로운 목적을 생성해 낼 수도 있다. 이 경우 새로운 목적에 대해 판단기가 적절한 실행 계획을 검색해 수행하여 실패 원인을 제거하겠지만 그렇다고해서 그 계획이 다시 수행됨을 보장하지는 않는다. 판단기는 실패 원인이 제거된 시점에서 검색된 실행 계획들 중에서 효율성이 더 좋은 계획이 존재하면 그것을 실행

시키게 된다. 경우에 따라서는 실패 원인을 제거하는 도중에도 당시 상황에서 실패했던 목표를 성취시킬 수 있는 실행 계획이 존재한다면 그것을 수행시킬 것이다. 따라서 이것은 항상 고정된 절차에 의해 수행되는 것이 아니라 당시 상황에 따라 적절한 대응을 취하는 것이기 때문에 반응형 로봇 행위를 지정하는 데 적합하다.

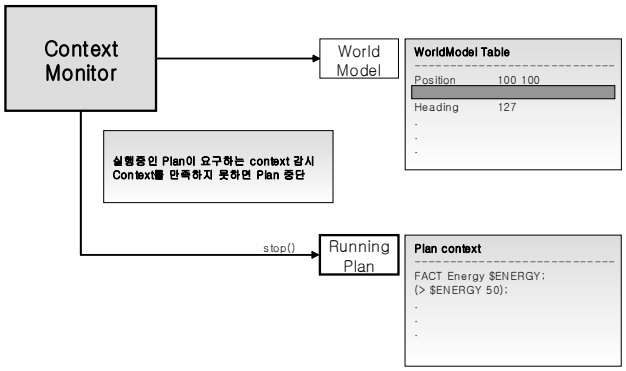

그림 4: 상황 감시기에 의한 실행중인 계획 감시

정리하자면 VivAce 에이전트는 그것이 처한 환경을 주기적으로 감지하면서 목적을 달성하기 위한 적절한 실행 계획을 검색하여 실행한다. 환경의 변화가 계획의 실행을 가로막으면 VivAce 에이전트는 적절한 대처를 통해 다른 계획을 검색해 실행시키거나 실패 원인을 분석해 그것을 제거하기 위한 목적을 생성한다. 이러한 과정을 통해 VivAce 에이전트는 다양한 환경의 변화에 대해 적절한 대처를 하면서 최종적으로 그것이 갖는 목적을 달성하게 된다.

3. VivAce 에이전트 적용 사례

본 논문에서는 로봇이 실제로 인간에게 서비스하는 상황을 가정하기 위해서 자바 3D 시뮬레이션 환경을 사용하였다. 이 환경에서 서비스를 받아야 하는 사람과 실제로 물이 들어있는 물통은 무작위로 생성되고 소멸되며 로봇이 이런 예측할수 없는 환경하에서의 행동을 결정하는데 있어서 에이전트 구조가 얼마나 효율적인가를 확인한다.

시뮬레이션 환경에서 구현한 부분은 크게 5 가지로 분류할 수 있다. 첫째 실제로 시뮬레이션 환경에 실제로 동작을 미치는 로봇, 둘째 VivAce 에이전트, 셋째 로봇에서 실세계모델의 정보를 입력하는 센서, 넷째 로봇을 움직이는 작동기(Actuator), 마지막으로 실제 로봇의 움직임을 담고 있는 계획이다.

```
public class RobotAgent extends VivAce {
   public RobotAgent(Agent agent) {
      this.addGoal("ACHIEVE ServeTheWater");
      this.assertFact("myPosition 0 0 0");
      this.addSensibleObject(new PositionSensor(this,actor));
      this.addPlan(new ServeTheGuestPlan(this,actuator));
      this.addExecutableObject(new GUIMonitor(this));
    …
    }
    …
```

위의 부분은 VivAce 에이전트의 본체이다. 실세계 모델에 초기 정보를 넣고 에이전트의 목표를 설정한다. 외부 환경을 감지할 감지기를 추가하고 에이전트가 수행할 수 있는 실행 계획을 등록시키는 모습을 볼 수 있다. 앞서 설명한대로 시나리오상의 구체적인 절차는 서술되지 않는다. 다수의 실행 계획에 대해 에이전트는 스스로의 목적을 위해 적절한 절차를 생성해 반응한다.

그림 5 는 시뮬레이터를 실행시킨 상황이다. 물통은 고정되어 있음을 가정하고, 가지고 있는 물이 다 떨어지면 무작위 위치에 다시 생성된다. 손님은 무작위로 움직인다.

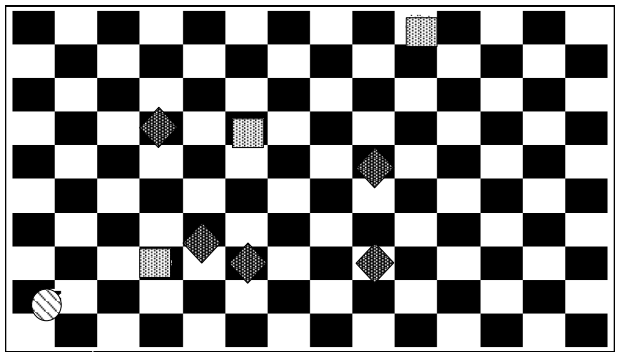

그림 5: 초기상태

화면 좌측 하단에 보이는 사선의 원이 서비스를 하는 로봇이고 마름모가 서비스 받아야 하는 손님이다. 사각형은 물을 공급하는 물통이다

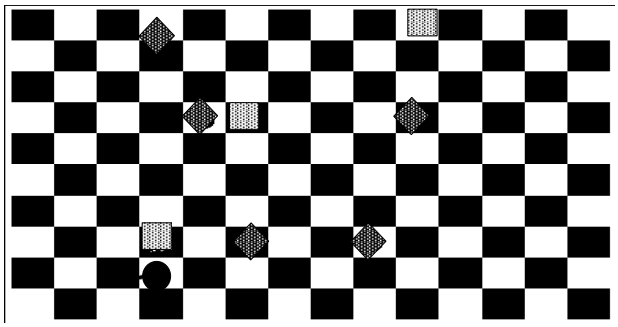

그림 6: 물을 공급받는 중

그림 6 은 최초 물을 갖고 있지 않은 서비스 로봇 에이전트가 물통에서 물을 공급받고 있는 상황이다. 물을 가지고 있는 로봇 에이전트는 사선이 아닌 검은색으로 표현된다.

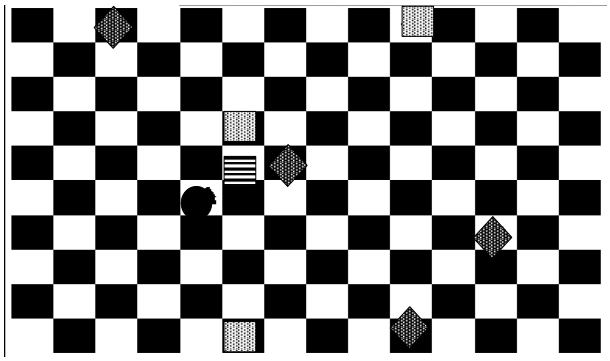

그림 7: 손님에게 물을 주는 중

그림 7 은 물을 가지고 있는 로봇 에이전트가 점으로 표현된 손님 에이전트로 가서 물을 공급하는 모습이다. 물을 공급받은 손님 에이전트는 수평줄무늬로 변한 후 사라진다.

본 적용 사례를 보면, 서비스 로봇의 VivAce 에이전트는 물통으로부터 물을 공급받는 계획, 손님에게 물을 서비스하는 계획 등이 제공되고 VivAce 에이전트는 상황 변화에 대해 적절한 대응을 해가며 손님에게 물을 제공한다.

4. 결론

본 논문에서는 다양한 변화가 존재하는 환경에 대해 적절한 대응을 하면서 에이전트의 목적을 성취하기위한 계획을 수행시키는 VivAce 에이전트를 소개하였다.

기존의 JAM 이 BDI 기반의 에이전트 시스템으로 검증되고 강력한 성능을 제공하기는 하였으나 병렬작업 지정과 기본 행위 지정에 제한이 있었다. 이에 JAM 을 기반으로 VivAce 라는 새로운 에이전트 구조와 언어를 개발하게 되었다.

VivAce 는 에이전트 시스템을 개발하는데 있어 네이티브 자바언어를 지원함으로써 기존의 에이전트 시스템 구축에 대해 익숙하지 않은 개발자들도 비교적 쉽게 에이전트 시스템을 개발할 수 있음을 보여주었으며 기존의 자바로 구현된 모든 리소스들을 어떠한 수정도 거치지 않고 바로 사용 함으로써 보다 에이전트 개발환경이 유연해지고 강력해 질 수 있음을 확인하였다. 아울러 쓰레드 기반의 계획 실행을 보장함으로써 환경 변화에 대한 반응 속도가 향상됨을 확인할 수 있었다.

또한 VicAce 는 위에서 언급한 장점이에도 Jade 와의 인터페이스와 온톨로지와의 인터페이스등 다중 에이전트 시스템을 개발하는데 필요한 많은 부분을 지원하고 있다. 이는 에이전트 시스템을 개발하는데 있어서 VicAce 가 편리하고 강력한 도구가 될 수 있음을 의미하는 것이다.

5. 감사의 글

본 연구는 과학기술특정연구개발사업의 인간기능 생활지원 지능로봇 기술개발사업단의 지원을 받았음.

참고문헌

[1] Stuart Russell and Peter Norvig. Artificial Intelligence: A Modern Approach. Prentice-Hall, 1995.

[2] Anand S. Rao and Michael P. Georgeff. BDI agents: From theory to practice. In Proceedings of the First International Conference on Multiagent Systems, pages 312-319, San Francisco, California, June 1995.

[3] Jaeho Lee, Marcus J. Huber, Edmund H. Durfee, and Patrick G. Kenny. UM-PRS: an implementation of the procedural reasoning systems for multirobot applications. In Conference on Intelligent Robotics in Field, Factory, Service, and Space (CIRFFSS'94), pages 842-849, Houston, Texas, March 1994.

[4] Nicholas Jennings, Katia Sycara and Michael Wooldridge. Roadmap of Agent Research and Development. Autonomous Agents and Multi-Agent Systems, vol. 1 pp. 7-38, 1998.

[5] http://mcis.jsu.edu/faculty/agarrett/jas3d/