지능형 로봇 제어를 위한 작업계획 생성기와 실행기의 통합☆

Integrating Task Planner with Plan Executive to Control Intelligent Robots

김 인 철* 김 하 빈** 신 행 철***
In-Cheol Kim Ha-Bin Kim Hang-Cheol Shin

요 약

본 논문에서는 지능형 로봇의 효율적인 제어를 위해 작업계획 생성기와 계획 실행기간의 연동방식을 제안한다. 이를 위해 본 논문에서는 먼저 작업계획 생성기와 실행기의 대표적인 유형과 연동방식들을 살펴본다. 그리고 BDI 구조의 계획 실행기에 주문형 작업계획 생성기를 연동하는 새로운 접근법을 소개하며, 이러한 연동방식에 기초한 지능형 로봇 제어구조를 설명한다. 또한 AIBO 로봇을 이용한 데모시스템의 구현을 통해 본 논문에서 제안한 이러한 연동방식과 제어구조의 유연성과 견고성을 분석해본다.

Abstract

This paper presents a way to integrate a task planner with a plan executive for efficient control of intelligent robots. For this purpose, it first surveys several types of the existing task planners and plan executives. And then it introduces a new approach that combines a BDI-based executive with an on-demand task planner, and it explains a control architecture of intelligent robots based upon this integrating paradigm. Finally it analyzes the flexibility and robustness of both the proposed integrating paradigm and the control architecture through implementation of a demo system using AIBO robots.

1. 서론

지능형 로봇의 제어구조는 로봇이 작업 목표 달성을 위해 센서 입력에 반응하여 어떻게 행동해야 하는지를 정의하는 것으로 볼 수 있다. 일반적으로 지능형 로봇의 제어구조는 복잡도를 고려하여 몇 개의 서로 다른 계층들로 나누어 구현되며, 각 계층은 각기 다른 레벨의 제어 문제를 다룬다. 일반적으로 제어구조의 최상위 계층은 작업(task) 단위의 제어를 다룬다. 이 계층에서는 로봇이 처한 현재 상황과 작업 목표를 고려하여 향후 수행할 작업계획을 수립하고 계획된 바대로 작업이 성공적으로 실행되고 있는지 모니터링하며, 필요한 경우 새로운 상황에 맞도록 재계획(replan)하는 기능들을 수행한다. 반복되는 일상적인 작업들이나 실시간성이 요구되는 작업들을 수행하기 위해서는 매번 작업계획을 새로 생성하는 것 보다 한번 실행에 성공한 적이 있는 계획을 기억하거나 수행할 계획을 미리 정의해주는 방법이 더 효과적이다. 그러나 실행하는 도중에 모든 계획들이 실패로 끝나거나 더 이상 적용할 새로운 계획이 없는 상황에 도달하면, 주문형 작업계획 생성기(on-demand task planner)의 도움을 받아 이러한 상황에 맞는 새로운 계획을 생성할 필요가 있다.

지능형 로봇을 위한 계획기반의 제어구조는 작업계획 생성기와 작업계획 실행기를 어떤 방식으로 구성하고 운영하느냐가 매우 중요한 설계 이

* 정 회 원 : 경기대학교 전자계산학과 교수
kic@kyonggi.ac.kr
** 준 회 원 : 경기대학교 대학원 전자계산학과(박사과정)
talkable@kyonggi.ac.kr
*** 준 회 원 : 경기대학교 대학원 전자계산학과(이학석사)
zest133@hanmail.net
[2007/03/05 투고 - 2007/03/09 심사 - 2007/03/28 완료]
☆ 본 연구는 2005학년도 경기대학교 학술연구비(연구그룹 연구과제)지원에 의하여 수행되었음

슈로 다루어지고 있다[1, 2, 3]. 그동안 우주항공 분야 등 여러 응용분야에서 지능형 로봇 제어를 위한 작업계획 생성기와 실행기간의 다양한 연동 방식들이 시도되어 왔다. 본 논문에서는 작업계획 생성기와 실행기의 개념 및 역할을 간략히 설명하고, 그동안 소개된 작업계획 생성기와 실행기의 대표적인 연동방식들을 살펴본다. 그리고 BDI 구조의 계획 실행기에 주문형 작업계획 생성기를 연동하는 새로운 접근법을 소개하며, 이러한 연동 방식에 기초한 지능형 로봇 제어구조를 설명한다. 또한 실제 AIBO 로봇을 이용한 데모시스템의 구현을 통해 본 논문에서 제안한 이러한 연동방식과 제어구조의 효율성을 분석해본다.

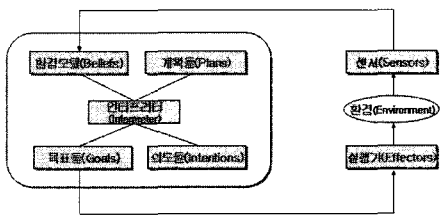
2. 작업계획 생성기와 실행기

2.1 개념 및 유형

일반적으로 작업계획이란 시간과 자원에 관한 제약을 만족하면서 주어진 작업 목표들을 달성할 수 있는 일련의 동작(action)들을 말한다. 그리고 추론을 통해 이러한 작업 계획을 수립해주는 소프트웨어 컴포넌트를 작업계획 생성기 혹은 작업계획기(task planner)라 부른다. 또 사전에 미리 수립해놓은 계획을 실제 작업 환경에서 실행해주는 소프트웨어 컴포넌트를 작업계획 실행기(plan executive)라 부른다.

전통적인 작업계획 생성기는 일반적으로 정적이고 모든 정보가 접근 가능한 이상적인 환경을 가정함으로써 계획을 수립하지만, 계획이 실행되는 실제 환경은 대부분 동적이고 잡음과 불확실성이 존재하는 불완전한 특성을 지닌다. 따라서 계획 실행기는 이러한 동적 실행 환경의 특성을 고려하여 계획 생성기가 제공하는 작업 계획보다 더 견고하고 풍부한 제어구조를 요구한다. 즉, 작업 실행기에 주어지는 대부분의 계획실행언어(execution language)에는 동작들의 순차구조(sequence) 외에 조건부 분기(conditional branch)와 반복구조(iteration) 등 다양한 실행 제어 구조들이 포함되어 있다. 또한 계획 실행기는 계획에 포함된 동작들을 차례대로 실행해주는 일 외에도 작업에 대한 계층적 분해(hierarchical decomposition)와 병행 동작들(concurrent actions)에 대한 조정, 자원 관리, 상태 모니터링, 장애 진단과 복구 등의 다양한 추가 기능도 수행해야 한다.

그동안 지능형 로봇 제어를 위해 개발되어온 작업계획 실행기들과 작업계획 생성기들을 유형별로 나누어 보면 다음과 같다. 먼저 별도의 자동화된 작업계획 생성기를 가지고 있지 않은 소위 실행만 가능한 시스템(execution-only system)들이 있다. 이들은 실행할 계획들이 모두 프로그래머들에 의해 수작업으로 작성되어 입력되는 것으로 가정한다. APEX[4], PRS[5], CRL[6]과 같은 실행기들이 이 유형의 대표적인 예가 된다. 또 다른 유형으로는 별도의 외부 작업계획 생성기와 결합된 실행 시스템(execution systems coupled with an external planner)들이다. TDL[7] RAP[8], RA[9], IDEA[10] 와 같은 실행기들이 대표적인 예들이다. 마지막 유형으로는 자체적으로 내부 작업계획 생성기를 포함하고 있는 실행 시스템(execution systems with internal planner)들이다. MPE[11], PROPEL[12], TITAN[13], RPL[14] 등은 내부 작업계획 생성기를 포함하고 있는 대표적인 실행기들이다. 앞서 살펴본 바와 같이 그 동안 지능형 로봇 분야나 우주항공 제어분야에 적용되었던 많은 작업계획 실행기들은 대부분 자체적으로 내부 작업계획 생성기를 포함하고 있거나 혹은 별도의 외부 작업계획 생성기와의 연동 기능을 제공하고 있다.



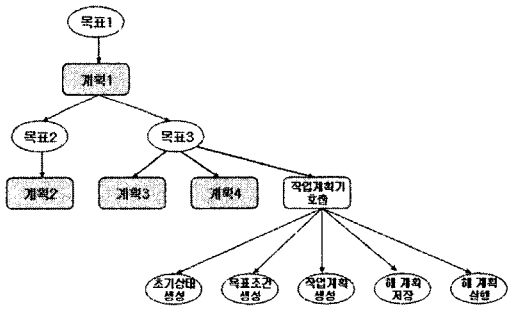
(그림 1) BDI 구조

2.2 연동방식

지능형 에이전트 연구분야에서 개발된 BDI (Belief-Desire-Intention) 구조는 동적 실행환경을 고려한 대표적인 에이전트 내부구조이다[5]. BDI 구조는 그림 1과 같이, 달성하고자 하는 목표(goal)들과 일반화된 계획(plan)들의 집합, 그리고 현재 실행 환경에 대한 모델(belief)을 바탕으로 매 순간 에이전트가 실행해야 할 동작들을 결정(intention)하고 실행하는 구조이다. BDI구조의 큰 특징 중의 하나는 작업 환경의 실시간성을 고려하여 새로운 목표가 주어질 때마다 매번 계획을 새로 생성하지 않는 대신 일반화된 계획들을 미리 저장하고 있다가 실행상황에 부합되는 계획들만을 선택적으로 적용해나감으로써 변화가 심한 동적환경에 신속히 대응할 수 있다는 점이다.

이러한 특성을 가진 BDI 구조는 실세계 환경에서 동작하는 지능형 로봇에서도 계획 실행기(plan executive)의 역할을 수행할 수 있다. 하지만 BDI 구조는 단순히 계획에 기술된 동작들을 차례대로 실행하는 실행기가 아니라, 그림 3과 같이 주어진 작업 목표와 실행상황을 고려하여 동적으로 실행계획을 선택하고 그 계획에 따라 다시 세부적인 부속 작업목표(subgoal)들로 분해하는 추론과정을 포함하고 있다. 따라서 BDI 구조는 실행환경에 따라 동적으로 구체적인 계획을 생성하고 실행한다는 의미에서 반응형 계획기(reactive planner)로도 볼 수 있다. 이러한 BDI 구조를 지능형 로봇 제어를 위한 작업계획 실행기로 이용하는 경우, 전통적인 작업계획 생성기는 BDI 실행기가 실행할 작업계획들을 미리 작성하여 공급하거나 혹은 실행 중에 필요에 따라 작업계획을 추가 생성해주는 주문형 작업계획기(on-demand task planner)로도 이용될 수 있다.

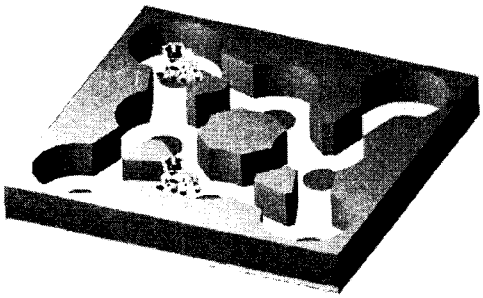
전자의 방식은 로봇 실행 환경이 비교적 정적인 경우에 효과가 높으며, 미리 작업계획 생성기를 통해 구체적인 작업계획을 생성해줌으로써 실행할 때 BDI 구조의 불필요한 추론과정을 줄일 수 있다. 후자의 방식은 BDI 구조에 내장되어 있는 기존 계획들만으로는 대처할 수 없는 실행상황이 발생할 때 큰 도움이 된다. 또한, 이 주문형 작업계획기 방식은 로봇 작업에 필요한 대부분의 작업계획들은 실행기 내부에 미리 저장하고 있으며, 필요한 경우에만 작업계획 생성기를 호출한다고 가정함으로써 실시간 제약이 존재하는 지능형 로봇 작업환경에 유리한 방식이다.



(그림 2) 주문형 작업계획기 연동방식

본 논문에서는 후자의 방식으로 결합되는 BDI 기반의 작업계획 실행기와 작업계획 생성기의 연동구조를 제시한다. 양자간의 연동을 위해서는 먼저 작업계획 실행 중에 주문형 작업계획기가 호출되어야 하는 시점을 결정하고, 그림 2와 같이 작업 목표를 달성하기 위한 BDI 계획들의 계층구조에 주문형 작업계획기 호출을 담당하는 특수한 하나의 BDI 계획을 추가한다. 이 BDI 계획은 주문형 작업계획기에 입력으로 넘겨줄 초기 상태(initial state)와 최종 목표상태(final state)를 구하는 단계, 외부의 작업계획 생성기를 호출하여 해 계획(solution plan)을 작성하는 단계, 작성된 작업계획을 실행하는 단계들로 구성된다. 따라서 BDI 기반의 작업계획 실행기는 이러한 연동구조를 기초로 실행 중에 예상하지 못한 새로운 작업 목표가 주어지거나 적용 가능한 기존의 작업계획들이 모두 실패로 끝난 상황을 만나면, 자동적으로 주문형 작업계획기 호출을 위한 특수한 작업계획이

실행되어 효과적으로 이러한 상황에 맞는 새로운 작업계획을 생성하고 실행할 수 있다.



(그림 3) 로봇의 작업환경

3. 지능로봇 제어구조

본 논문에서는 앞서 제안한 작업계획 생성기와 실행기의 연동구조를 기초로 지능로봇 제어구조를 설계하고 구현하였다. 이 절에서는 본 연구에 이용된 로봇 플랫폼과 작업환경을 소개하고, 그 위에 구현된 계획기반의 제어구조에 대해 자세히 설명한다.

3.1 로봇 플랫폼과 작업환경

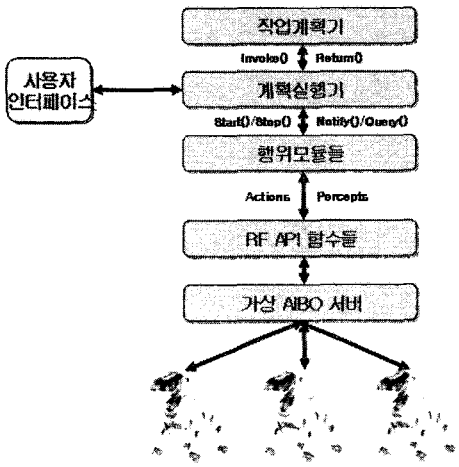
본 연구에서는 Sony사에서 개발된 애완용 AIBO 로봇을 기본 플랫폼으로 이용하였다. 이 AIBO 로봇은 64비트 RISC 프로세서와 64MB 메모리를 가지고 있으며, 각각 3개의 관절을 포함한 4개의 다리를 가지고 있다. 또 이 로봇플랫폼은 시각인식을 위한 CMOS 컬러 이미지 센서를 비롯하여, 장애물과의 거리측정이 가능한 적외선 센서 등 다수의 센서들을 몸에 부착하고 있으며, 스피커와 마이크로 폰을 통해 소리를 발생시키거나 들을 수 있다. 또한 얼굴에는 감정표현이 가능한 LED패널을 가지고 있고, 몸통부에는 무선 랜 통신기능을 내장하고 있다. AIBO ERS-7M3의 모든 서비스 프로그램들은 실시간 운영체제인 Aperios 위에서 실행되며, 개발자를 위한 프로그래밍 환경으로 C++ 기반의 Open-R과 Remote Framework [15] 등을 제공하고 있다.

이러한 AIBO 로봇을 이용하여 이루고자 하는 작업들은 대부분 그림 3과 같은 복잡한 3차원 미로 환경에서 사용자가 지정해주는 임의의 목적지까지 장애물을 피해 최단경로로 찾아가는 일들이다. 로봇이 이러한 임무를 성공적으로 수행하기 위해서는 미지의 공간 내에 자신이 처음 놓여진 위치를 파악하는 일, 패턴인식을 통해 이동 가능한 이웃 경로점(way point)들을 찾아내는 일, 자세를 바꾸거나 몸을 돌려 이동을 준비하는 일, 선택된 경로점까지 장애물을 피해 걸어가는 일, 이동 중에도 계속해서 목표 지점을 놓치지 않고 시선을 유지하는 일 등 다양한 지능행위들이 요구된다.

3.2 제어구조

AIBO 로봇을 위한 전체 제어구조는 그림 4와 같이 하나의 계층구조를 이룬다. AIBO 로봇의 세부적인 모션과 센서 데이터를 다루는 제어부의 최하위 계층은 Sony AIBO Remote Framework 층이다. Remote Framework층은 다시 공통의 Virtual AIBO Server와 이에 대한 서비스를 요청하는 Remote Framework API층으로 구성된다. 이 계층 위에 본 연구에서 제시하는 계획기반의 제어부가 위치한다. 먼저 맨 아래에 로봇의 행위모듈 층이 존재하는데, 이 계층은 Remote Framework API를 이용하여 C++로 구현된 로봇의 다양한 행위모듈들로 구성된다.

AIBO 로봇을 위한 유용한 행위들은 수없이 많이 존재할 수 있으나, 본 연구에서는 앞서 설명한 작업환경에 필요한 행위들로만 한정하여 구현하였다. AIBO 로봇의 주된 작업들은 사용자가 지정해주는 목적지까지 최단경로를 계획하고 이동하는 작업들이 대부분이다.



(그림 4) 계획기반의 로봇제어구조

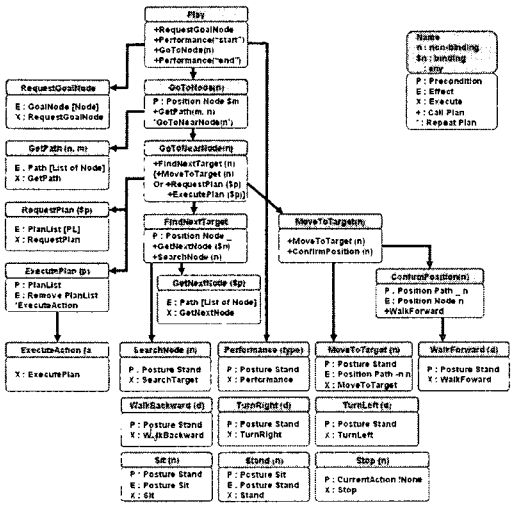
이를 위해 로봇은 바닥에 특별한 도형패턴으로 표시된 이동 경로점(way point)들을 실시간 시각 인식을 통해 찾고 이들을 기초로 환경 맵을 작성하고 그래프 탐색기법을 적용하여 목적지까지 최단경로를 계획하여야 한다. 각 경로점을 표시하는 패턴들은 AIBO 로봇의 패턴학습 기능을 이용하여 작업 실행 이전에 미리 학습시켜둠으로서 실행 중에 효과적으로 이들을 인식할 수 있도록 하였다.

행위모듈 계층 위에는 행위 모듈 각각의 실행을 제어하는 작업계획 실행기 층이 위치한다. BDI 기반의 작업계획 실행기는 각 행위모듈의 실행과 중지에 대응하는 기본 작업계획들뿐만 아니라 이들을 결합하여 보다 복잡한 상위의 작업 목표를 달성해주는 다양한 복합 작업계획들을 미리 내장하고 있으며, 상황에 맞게 이들을 선택하여 실행하는 역할을 수행한다. 계획기반 제어구조의 최상위 계층은 주문형 작업계획기 층으로서, 실행 중 계획 실행기의 요청이 있을 때에 필요한 새로운 작업계획을 즉시 생성해주는 역할을 수행한다. 끝으로 사용자와 로봇간의 혼합형 주도권 제어를 지원하는 그래픽 기반의 사용자 인터페이스가 제어시스템의 한 부분을 이룬다.

3.3 작업계획 생성과 실행

본 연구에서는 Michigan 대학교에서 개발된 대표적인 BDI 구조인 UM-PRS[16]를 AIBO 로봇 제어를 위한 계획 실행기로 채용하였다. UM-PRS는 높은 목표-지향성과 환경변화에 빠른 반응성을 보이는 지능형 에이전트 구조이다. UM-PRS의 인터프리터는 미리 정의된 일반화된 계획들과 달성하고자 하는 목표, 그리고 환경과 로봇의 상태정보를 내부에 유지하면서, 매 순간 가장 적합한 계획을 결정하고 이것을 실행한다. 현재 실행중인 계획이라도 환경의 상태나 목표 등의 변화가 있으면 언제든지 실행을 중단하고, 다른 계획을 찾아본다. UM-PRS 에서 하나의 계획은 하나의 프로시듀어(procedure)로 기술되는 본체(body)와 계획이름(name), 목표(purpose), 적용조건(context), 효과(effect), 실패 처리부(failure), 우선순위(priority) 등으로 표현된다. 또 하나의 UM-PRS 계획은 본체에 다시 여러 개의 부속 목표(subgoal)들을 포함할 수 있도록 허용하고 있다. 따라서 UM-PRS의 계획실행 과정은 최상위 작업 목표를 하위의 작은 세부 작업 목표들로 나누어 가면서 각 작업 목표에 적합한 구체적인 계획들을 결정하고 실행하는 것으로 요약할 수 있다.

작업계획 실행기인 UM-PRS에 내장되는 작업계획들은 크게 2 가지 종류로 나눌 수 있다. 첫 번째 유형은 하위 계층에 속한 각 행위모듈의 실행을 시작하거나 멈추는 역할을 하는 기본 작업계획(primitive plan)들이다. 두 번째 유형은 이와 같은 기본 계획들을 기초로 보다 상위의 작업 목표를 달성하기 위한 복합 계획 (composite plan)혹은 추상 계획(abstract plan)들이다. 따라서 AIBO 로봇 제어를 위한 작업계획들 간의 관계는 그림 5와 같이 계층구조를 이룬다. 또한 작업계획들의 집합에는 앞서 설명한 바와 같이 실행 중 주문형 작업계획 생성기와의 연동을 위해 주문형 작업계획기 호출을 위한 특별한 UM-PRS 계획들을 포함하고 있다.



(그림 5) 작업계획들의 계층구조

한편, 본 연구에서는 Macedonia 대학교에서 개발한 GRT[17]를 AIBO 로봇을 위한 작업계획 생성기로 이용하였다. GRT는 FF[18]나 HSP[19] 등과 같이 상태공간 상의 탐색을 통해 작업계획을 생성하는 대표적인 휴우리스틱 계획 생성기(heuristic planner)이다. GRT의 계획생성과정은 전처리 단계(pre-processing phase)와 탐색단계(search phase)로 이루어진다. GRT는 오프라인 전처리작업을 통해 계획문제로부터 각 사실(fact)과 목표(goal)사이의 추정거리를 계산해내고 이들을 하나의 테이블에 저장해둔다. 그리고 해 계획을 구하는 탐색과정동안에 이러한 사실과 목표사이의 추정거리를 토대로 탐색공간 상의 각 상태에서 목표까지의 추정거리를 계산해냄으로써 효과적으로 최적-우선 탐색(best-first search)을 전개한다. 작업계획 생성기 GRT를 이용하여 효율적으로 작업계획을 생성하기 위해서는 수행 가능한 로봇 동작들에 대한 정의를 미리 표준 언어인 PDDL (Planning Domain Description Language)로 기술하여 내장하고 있어야 한다. 이를 위해 본 연구에서는 앞서 소개한 각각의 행위모듈을 하나의 PDDL 동작으로 정의해주었다. 각 PDDL동작은 STRIPS의 전통을 따라 전-조건(precondition)과 효과(effect)로 이루어진 하나의 쌍으로 표현된다. 그림 6은 로봇의 이동 동작인 move의 PDDL 정의를 보여주고 있다. 효율적인 계획 생성을 위해 로봇 동작들에 대한 정의는 미리 내장하고 있는 반면에, 실제 계획문제(planning problem)들은 계획 실행기의 요청에 의해 필요할 때마다 즉각적으로 만들어진다.

3.4 실행 모니터링과 사용자 직접 제어

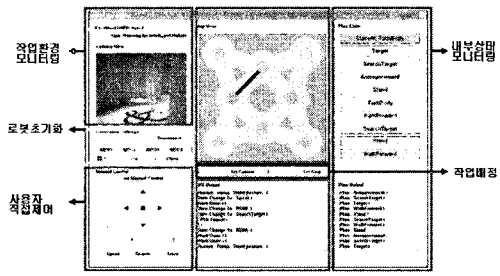
사용자가 효과적으로 로봇에 작업을 부여하고 실행을 모니터링하기 위해서는 편리한 사용자 인터페이스를 제공하여야 한다. 사용자는 이와 같은 사용자 인터페이스를 통해 로봇을 초기화하고 새로운 작업을 부여할 수 있어야 할 뿐 아니라, 로봇의 판단과 행동을 이해하기 위해서는 로봇이 감지하는 실시간 센서정보와 이에 따른 내부 계획의 생성과 실행과정도 모니터링할 수 있어야 한다. 또한, 실행 중인 로봇에 예기치 못한 사고나 오동작이 발생하면 사용자는 인터페이스를 통해 로봇을 긴급히 멈추거나 안전한 상황이 될 때까지 사용자가 직접 제어할 수 있어야 한다.

```
(:action move
 :parameters (?aibo ?source ?destination)
 :precondition
        (and
                (node ?source)
                (node ?destination)
                (link ?source ?destination)
                (link ?destination ?source)
                (on ?aibo ?source)
                )
 :effect
                (and
                        (not (on ?aibo ?source))
                        (on ?aibo ?destination)
                )
)
```

(그림 6) PDDL로 표현된 동작의 예

이와 같은 기능적 목적을 위해 음성이나 제스처 등 다양한 형태의 사용자 인터페이스를 이용할 수 있으나, 내부 정보의 실시간 브라우징과 사용자 직접 제어를 위해 본 연구에서는 그림 7과

같은 그래픽 기반의 사용자 인터페이스를 설계하고 구현하였다. 화면의 좌측 상단에는 로봇의 카메라에 들어오는 영상과 인식된 경로점 패턴을 보여주고 있으며, 그 아래에는 로봇을 초기화해주는 제어부가 위치한다. 화면의 상단 중심부에는 맵상에 현재 진행하고 있는 경로를 나타내고 있으며, 그 아래에는 새로운 작업(목적지)을 지정해주는 제어부가 위치하며, 우측 상단 및 하단에는 선택된 로봇 내부의 작업계획과 상태정보를 보여주고 있다. 화면의 하단 좌측에는 사용자가 로봇의 동작을 직접 하기 위한 사용자 제어부가 위치하고 있다.



(그림 7) 사용자 인터페이스 화면

작업과 연관된 로봇의 자율행위는 대부분 앞서 소개한 계획기반의 제어구조에 의해 결정되고 실행된다. 즉, 로봇 내부의 주문형 작업계획기와 작업계획 실행기의 연동을 통해 로봇 스스로 작업목표를 설립하고 이에 필요한 새로운 작업계획들을 생성하고, 또 상황에 맞게 이들을 선택적으로 실행함으로써 주어진 작업들을 효과적으로 달성해간다. 하지만 로봇의 불완전한 인식능력과 기계동작의 잠재적 오류를 고려한다면 로봇의 자율행위에만 의존하여 위험성을 내포한 작업을 수행하는 것은 바람직하지 않다. 따라서 많은 실용 로봇들은 아직도 사용자의 원격제어(remote control)에 전적으로 의존해 작업을 하거나 부분적으로 사용자의 감시와 직접 제어를 허용하는 혼합 주도권 제어(mixed-initiative control) 방식으로 작업을 수행한다.

본 연구는 사용자 인터페이스와 내부 작업계획들을 이용하여 제한적인 혼합 주도권 제어 메커니즘을 구현하였다. 본 연구에서 구현된 혼합 주도권 제어는 주로 로봇이 오동작을 발생하는 응급상황에 대처하기 위한 것이다. 거의 모든 로봇들에는 응급상황에 대처하기 위해 긴급정지 기능을 제공하고 있다. 하지만 대부분의 경우는 일단 긴급정지 기능이 이루어지면 그와 함께 로봇의 자율제어는 거기서 끝나버리고 이후의 제어는 모두 사용자에게 맡겨진다. 본 연구에서는 응급상황에 대한 긴급정지 기능과 사용자 직접 제어 기능을 제공하지만 일단 사용자 직접 제어가 끝나면 언제든지 다시 계획기반의 자율 제어 체제로 복귀할 수 있도록 설계하였다. 이를 위해 사용자의 직접 제어명령을 받고 이를 실행하는 것도 내부적으로는 기존의 작업계획 체제를 이용하여 처리되도록 하였다.

```
CYCLE
{
        RETRIEVE currentAction $action;
        EXECUTE Update_Action $action;
        UPDATE (currentAction)(currentAction $action);

        RETRIEVE currentAction $action;
        EXECUTE UpdateUserControlMode $state;
        WHEN: TEST (&& (== $state "TRUE")(!(== $action "NONE")))
        {
                UPDATE (inUserControlMode)(inUserControlMode "TRUE");
        };

        RETRIEVE inUserControlMode $currentState;
        WHEN: TEST (== $state "FALSE")
        {
                UPDATE (inUserControlMode)(inUserControlMode "FALSE");
        };

}
```

(그림 8) 사용자 직접 제어를 위한 반복적 계획

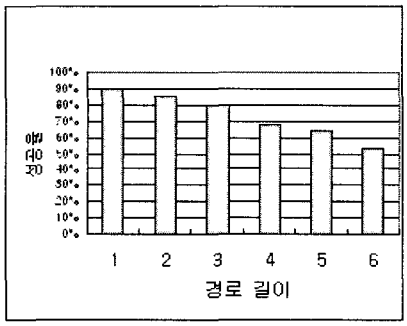
그림 8은 사용자 직접 제어 명령을 처리하기 위한 반복적 계획(cyclic plan)의 예를 보여주고 있다. 이 계획은 현재 실행 중인 작업계획의 각 스텝이 끝날 때마다 한번 실행되는 특수한 계획으로서, 주기적으로 사용자 직접 제어 모드인지를 체크하고 사용자 명령이 있으면 이것을 받아 처리하는 역할을 수행한다. 만약 자율행위 모드에서 하나의 작업계획이 수행 중일 때, 사용자 인터페이스를 통해 사용자 직접 제어 요청이 주어지면 실행 중이던 작업계획의 현재 스텝이 완료될 때

까지 기다렸다가 이 반복적 계획으로 제어가 옮겨져 사용자 직접 제어 명령들을 처리하게 되며, 이들에 대한 처리가 끝나고 나면 다시 원래 실행중이던 작업계획의 다음 스텝으로 실행제어가 되돌아간다. 따라서 이것은 사용자 직접 제어 이후에 다시 원래의 로봇 자율제어 체제로 정상적으로 복귀할 수 있음을 의미한다.

4. 실험

본 연구에서는 앞서 제안한 작업계획 생성기와 실행기의 연동방식에 기초한 지능로봇 제어구조가 실세계 작업들에서 얼마나 견고한 성능을 보이는지를 알아보기 위한 실험을 실시하였다. 실험환경은 2㎡ 크기의 정사각형 판 위에 8개의 서로 다른 패턴을 가진 바닥 표식이 존재하고 인접한 표식 사이의 통로를 제외한 부분은 벽으로 이루어져 있다. 사용자는 로봇의 초기 위치와 최종 목적지를 지정해줌으로써 로봇이 수행해야 하는 작업을 로봇에게 전달할 수 있으며, 또한 중간 경유 노드를 추가함으로써 더 복잡한 작업을 지정해줄 수도 있다. 로봇은 초기위치에서 중간 경유 노드를 거쳐 최종 위치까지 최단 거리로 이동하게 되는데 이 때 가능한 경로의 수는 1에서 6까지 이다. 작업 계획을 수행하는 동안 물리적인 충돌로 인해 더 이상 작업 수행이 불가능할 경우와 사용가능한 작업 계획으로 더 이상 진행되지 못할 경우, 그리고 자신의 위치를 잃어버린 이후 다시 위치를 찾지 못하거나 잘못된 위치로 인식할 경우, 작업 수행 실패로 간주 하였다.

그림 9는 약 170회 시행한 실험의 결과를 요약한 것으로, 경로 길이의 증가에 따른 평균 작업 성공률을 나타내고 있다. 실험환경에서 출발점에서 목적지까지 경로의 길이는 작업의 난이도를 표현하는 하나의 척도로 볼 수 있다. 로봇의 불완전한 인식 능력과 기계동작의 잠재적 오차로 인해, 이동하여야 할 경로 수가 증가함에 따라 작업 성공률은 점차적으로 감소하였다. 하지만 경로 길이가 3까지는 약 86%가 넘는 높은 성공률을 보였고, 가장 난이도가 높은 경로길이 6의 경우에도 약 53%의 성공률을 보였다. 실험한 이용한 로봇의 기본 인식모듈과 행위모듈의 낮은 성능에도 불구하고, 이와 같은 높은 작업 성공률을 보인 것은 이들을 상위계층에서 관리하고 제어하는 계획기반 제어구조의 유연성과 견고성을 입증한 결과로 평가할 수 있다.



(그림 9) 작업 성공률

5. 결론

본 논문에서는 주문형 작업계획 생성기와 실행기의 연동을 통해 지능형 로봇의 자율행동을 제어하는 계획기반의 제어구조를 개발하였다. 그리고 미로 속에서 경로를 계획하고 이동하는 지능로봇에 이 제어 구조를 적용하였고, 실험을 통해 이 제어구조의 유연성과 견고성을 분석해보았다. 본 논문에서 제시한 작업계획 생성기와 실행기의 연동방식은 작업환경의 실시간성을 만족시킬 수 있을 뿐 아니라 BDI 구조가 갖는 표현력과 다양한 실행제어 기능을 충분히 이용할 수 있다는 장점을 가진다. 또한 본 논문에서 제시한 로봇과 사용자간의 혼합 주도권 제어 방식은 위험성과 난이도가 높은 작업들에서 매우 효과적으로 적용될 수 있다. 현재는 응급상황 대처에만 주로 적용되고 있는 혼합 주도권 제어를 작업계획을 세우고 실행하는 로봇 작업 전반으로 확장해보는 향후 연구도 가능하다.

참 고 문 헌

[1] Gallien, G. and Ingrand F., "Planning and Plan Execution for Autonomous Robots," Proc of RFIA-06, 2006.

[2] Garcia-Martinez, R. and Borrajo, D., "An Integrated Approach of Learning, Planning, and Execution," Journal of Intelligent and Robotic Systems, vol.29, pp.47-78, 2000.

[3] Haigh, K. Z. and Veloso, M. M., "Planning, Execution and Learning in a Robotic Agent," Proc. of the 4th Int. Conf. on Artificial Intelligence Planning Systems, pp.120-127, 1998.

[4] Freed M., "Managing Multiple Tasks in Complex, Dynamic Environments," Proc. of the AAAI-98, 1998.

[5] Rao, A. and Georgeff, M., "BDI-agents: from Theory to Practice," Proc. of the 1st Int. Conf. Multiagent Systems, 1995.

[6] Bresina J.L. and Washington R., "Robustness via Runtime Adaptation of Contingent Plans," Proc. of the AAAI-2001 Spring Symposium: Robust Autonomy, 2001.

[7] Simmons, R. and Mitchell, T. M., "A Task Control Architecture for Mobile Robots," Working Notes of the AAAI Spring Symposium on Robot Navigation, 1989.

[8] Firby J., "Adaptive Execution in Complex Dynamic Domains," Ph.D. Thesis, Yale University Technical Report YALEEU/CSD/RR #672, 1989.

[9] Pell B., et. al., "A Remote Agent Prototype for Spacecraft Autonomy," Proc. of the SPIE Conf. on Optical Science, Engineering, and Instrumentation, 1996.

[10] Muscettola N., et. al., "IDEA: Planning at the Core of Autonomous Agents," Proc. of the AAAI-2001, 2001.

[11] Barrett A., Knight R., Morris R., Rasmussen R., "Mission Planning and Execution within the Mission Data System," International Workshop on Planning and Scheduling for Space, 2004.

[12] Levinson R., "Unified Planning and Excution for Autonomous Software Repair," Proc. of ICAPS 2005 Workshop on Plan Execution: A Reality Check, 2005.

[13] Williams B.C. et. al., "Model-Based Programming of Fault-Aware Systems," AI Magazine, Vol.24, No.4, pp.61-75, 2004.

[14] Beetz, M., Plan-Based Control of Robotic Agents: Improving the Capabilities of Autonomous Robots, Springer, 2002.

[15] SONY, AIBO Software Development Environment(SDE), http://openr.aibo.com, 2006.

[16] Lee J., Huber, M., Durfee, E., and Kenny, P., "UM-PRS: An Implementation of the Procedural Reasoning System for Multirobot Applications," Proc. of the Conf. Intelligent Robotics in Field, Factory, Service, and Space, pp.842-849, 1994.

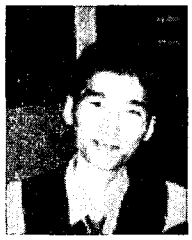
[17] Refanidis, I. and Vlahavas, I., "The GRT Planner: Backward Heuristic Construction in Forward State-Space Planning," Journal of Artificial Intelligence Research, vol.15, pp.115-161, 2001.

[18] Hoffmann, J. and Nebel, B., "The FF Planning System: Fast Plan Generation Through Heuristic Search," Journal of Artificial Intelligence Research, pp.253-302, 2003.

[19] Bonet, B. and Geffner, H., "Planning as Heuristic Search," Journal of Artificial Intelligence, vol.129, no.1, 2001.

◐ 저 자 소 개 ◑

김 인 철(In-Cheol Kim)
1987년 서울대학교 대학원 계산통계학과(이학석사)
1995년 서울대학교 대학원 계산통계학과(이학박사)
1996년~현재 경기대학교대학교 전자계산학과 교수
관심분야 : 인공지능, 지능로봇시스템, 자동계획 및 기계학습.
E-mail : kic@kyonggi.ac.kr



김 하 빈(Ha-Bin Kim)
2002년 경기대학교 전자계산학과(이학사)
2004년 경기대학교 대학원 전자계산학과(이학석사)
2004년~현재 경기대학교 대학원 전자계산학과(박사과정)
관심분야 : 지능로봇시스템, 에이전트구조, 게임인공지능, 기계학습
E-mail : talkable@kyonggi.ac.kr

신 행 철(Hang-Cheol Shin)
2004년 경기대학교 전자계산학과(이학사)
2006년 경기대학교 대학원 전자계산학과(이학석사)
관심분야 : 인공지능, 자동계획시스템, 시맨틱 웹, 의료정보시스템
E-mail : zest133@hanmail.net